Kernel-Based Sampling of Arbitrary Signals

Simone Cammarasana, Giuseppe Patanè

Consiglio Nazionale delle Ricerche
Istituto di Matematica Applicata e Tecnologie Informatiche
Genova, Italy

Abstract

Point sampling is widely used in several Computer Graphics' applications, such as point-based modelling and rendering, image and geometry processing. Starting from the kernel-based sampling, which approximates an input signal on a regular grid as the sum of Gaussian kernels, we introduce a set of additional variables that control the kernels' width and height. These additional variables allow us to improve the quality of the distribution of the samples, and to achieve a higher approximation accuracy and a more accurate feature preservation, with a slightly higher computational cost. To further improve the sampling with respect to the input data, we introduce a *sampling initialisation* for processing high resolution signals, without incurring in limits for memory allocation, and a *sampling optimisation*, which adaptively selects the number and location of the samples to achieve the target approximation accuracy, without oversampling the input signal. To show the generality of the proposed approach for unstructured data of arbitrary dimension, we apply our kernel-based sampling to different types of data, such as 2D images, solutions to PDEs on arbitrary domains, and vector fields.

Keywords: Kernel-based sampling, multi-scale kernel-based sampling, adaptive kernel-based sampling, data and signal sampling, signal approximation, radial basis functions

1. Introduction

Point sampling is widely used in several Computer Graphics applications, such as point-based modelling [41] and rendering [47], image and geometric processing [43, 35]. Point sampling is strictly related to image half-toning [30], which consists in simulating the full tone range of an image through a proper pattern of dots. This problem was raised with earlier mechanical printers, in order to reproduce the photographs on newspapers, using only one colour of ink for recreating a 256 grey level image. The image is replicated by placing more points in darker areas, according to local patterns, geometries, and colour intensities. The technical evolution of modern printers has achieved very good results in replicating continuous-tone images; for example, a modern laser printer can reach the resolution of 3K Dots per Inch. Point sampling has been applied also to image reconstruction [55], anti-aliasing [26], dithering [21], QR codes reconstruction [10], and artistic visualisation [36].

Important aspects of point sampling are *adaptation* to the input signal in order to guarantee that the sampling density is proportional to the image density or to the complexity of the signal in a given area; *feature preservation* without either over-smoothing or artifacts in the sampling or in the reconstructed signal; and *spectral properties* (e.g., blue-noise property) that allow us to achieve visually superior images, as the distribution of photoreceptors in a primate eye possesses the blue-noise characteristic [58].

Overview and contribution. Our starting point is the kernel-based sampling [64], which approximates an input signal on a regular grid (e.g., a 2D or 3D image) as the sum of Gaussian kernels, whose centres are computed through the minimisation of an energy functional. The kernels have a fixed size and the same energy, in order to provide the same weight to the energy functional. This choice generates artifacts in the sampling of images with complex patterns, and limits the approximation accuracy and the distribution of the samples in case of irregularly distributed data.

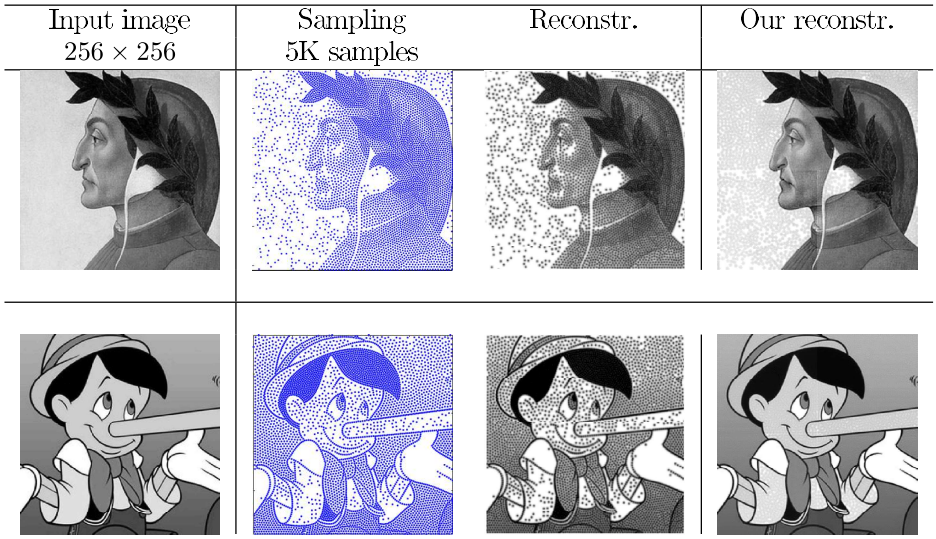


Figure 1: Input images, samples, and reconstructions induced by the original and the proposed kernel-based sampling, which better preserves sharp features and variation of grey levels.

To visualise these limitations, we have applied the kernel-based sampling to two images with complex features and variation of grey levels (Fig. 1). The sampling results (i.e., the centres of the kernel functions) and the image reconstruction correctly preserve the geometries and the features of the input image. However, some artifacts are still visible on Dante's background and on Pinocchio's nose, due to a non-optimal selection of the kernel width, and to almost empty intersections of the kernel supports.

Starting from the kernel-based sampling (Sect. 2), we introduce a set of additional variables, which allow us to improve the quality of the kernel-based sampling in terms of the distribution of the samples, and to further reduce the approximation error. More precisely, we introduce the $(\boldsymbol{\sigma}, \boldsymbol{\alpha})$ *kernel-based sampling* (Sect. 3), where $\boldsymbol{\sigma}$ controls the kernel width and $\boldsymbol{\alpha}$ is the vector of coefficients that express the input signal as a linear combination of the Gaussian kernels. To further improve the flexibility of the kernel-based sampling with respect to the input data, we introduce a *sampling initialisation* for processing high resolution signals, without incurring in limits for memory allocation, and a *sampling optimisation*, which adaptively selects the number and location of

the samples in order to achieve the target approximation accuracy, without oversampling the input signal (Sect. 4)

As main contributions with respect to previous work on samplings, we improve the sampling quality and the approximation accuracy, also achieving a more accurate feature preservation, with a slightly higher computational cost related to $2n$ additional variables, where n is the number of samples. Analogously to Gaussian Mixture Models (GMMs), we approximate the input signal as a linear combination of radial basis functions (RBFs), whose parameters are optimised through the minimisation of an energy functional. As main novelties and improvements with respect to GMMs, we apply our kernel-based sampling to structured and unstructured data with an arbitrary dimension, i.e., our method is independent of the dimensionality and spatial organisation of the input data. In contrast, GMMs have been applied to signals defined on regular grids. To show this generality, in the experimental tests we have applied our sampling to 2D/3D images, to the solutions to PDEs on arbitrary 2D/3D domains, and to vector fields (Sect. 5). Furthermore, we consider a global instead of a local optimisation of the variables, which is based on a global minimisation of the energy functional. Analogously to GMMs, a kernel-based sample corresponds to the mean of the Gaussian function in 1D, the kernel width is equivalent to the standard deviation, and the kernel weight corresponds to the kernel scale.

Finally (Sect. 6), we describe the main limitations of our work and future research.

2. Related work

We briefly review previous work on signal sampling, according to six main classes: physics-based sampling, Gaussian mixture models and kernel-based sampling, stochastic sampling, kernel-based image sampling, tessellation-based sampling, and optimal transport.

Physics-based sampling is driven by physical equations, such as fluid dynamics, engineering, and electromagnetism. In [48], the particles (i.e., samples) are placed through a model inspired by electrostatics. The particles' attraction

and repulsion are governed by the Coulomb laws, and the optimisation of the particles' position leads to an electrostatic equilibrium of the system. In [22], the particles' density is optimised by solving the Lagrangian formulation of the governing equations of compressible flow. In [18], the principles of natural selection acting on biological organisms are applied to the definition of a genetic algorithm for sampling, where the population (i.e., the samples) evolves with crossover and mutation events, until an optimal solution is reached. In [4], the interaction among particles is simulated through mechanical laws, including particles' interaction, and neighbouring effects. In [42], the mesh is optimised by solving a static force equilibrium in a truss structure, where the edges of the grid correspond to bars and the points correspond to the joints of the truss. The L_p-Gaussian kernels [62] reproduce the inter-particle energy to set-up the sampling patterns, and the surface is reconstructed exploiting the distribution of the particles. In [63], the input anisotropic mesh is transformed into a high-dimension isotropic space; then, the mesh vertices are computed through the optimization of an energy functional. The sampling of images dominated by low frequencies [25] exploits the link between Fourier analysis and spatial statistics, iteratively applying a force to the samples that depends on the geometry of the samples, without generating artifacts (e.g., aliasing).

Gaussian mixture models (GMMs) approximate an input signal as a mixture of probability distributions and have been successfully applied to different research problems such segmentation [23], denoising [54], inverse problems [61], inpainting [60], registration [46]. GMMs are also relevant for clustering [3], where the most challenging aspects are the variables' selection [33], initialisation [57], and partitioning [52]. Finally, GMMs are ubiquitous in engineering [29], chemistry [13], biomedical [45], and signal processing [59].

In the *kernel-based sampling* [50], the samples are associated with kernel functions, whose linear combination with constant coefficients approximates the input signal. In [20], the input signal is approximated in order to generate a point set with blue-noise properties. Each kernel function has a predefined support, and a statistical model is defined to allow solutions that further reduce

the minimum of the energy functional. In [64], Gaussian kernels with a fixed support are applied to sample the input image, through an optimisation algorithm, whose variables are the samples' position. In [37], the optimal sampling conditions are obtained by combining spectral analysis with kernel functions, and the samples are used to reconstruct a continuous surface with the desired smoothness. In [6], super-resolution images are reconstructed by applying a convolution operation and assuming a similarity among correlated neighbours. In [7], a sampling method with blue-noise properties is defined by considering both spatial and non-spatial properties and by modulating the samples' position with a domain-independent similarity. In [24], Gaussian kernels are used for approximating a continuous probability density function.

Stochastic sampling applies a probabilistic approach to generate a sampling where points are tightly packed, with a minimum distance constraint. Dart throwing [11] places the samples sequentially; if the new sample does not satisfy the constraint, then it is rejected. This method can be applied to surface sampling [5] and image rendering [15]. In the hierarchical approach of the dart throwing algorithm [56], the domain is subdivided in quad-tree regions, and the samples are placed only at active squares (i.e., where a sample is not already present). In [17], a very efficient algorithm for generating Poisson-disk distributions is achieved by representing the available neighbourhood for the insertion of a new sample through a data structure called scalloped region.

Kernel-based image sampling optimises the approximation accuracy of the input image, through the minimisation of an energy functional. In [38], the input image is compared with its sampling according to three metrics: luminance, contrast, and structure, integrated with the tone similarity. In [19], a perceptually-based approach is proposed for progressive rendering, where samples are added through an iterative refinement, by computing the distance between the original image and its approximation with contrast maps corresponding to spatial frequency bands [39]. Constraints on the edge [31] and greyscale [1] preservation are also added to improve the quality of the image sampling.

Tessellation-based sampling methods apply a centroidal Voronoi tessellation,

whose vertices provide a sampling of the input domain. The Lloyd's method [32] computes the centres of the tessellation through an iterative algorithm, where the vertices are updated by computing the mass centroid of the Voronoi regions. In [14], the Lloyd's algorithm is specialised to half-toning applications. In [2], a variant of Lloyd's method is proposed by imposing that each point has the same capacity, which is defined as the area of the related Voronoi region, weighted with a density function. This method enhances the blue-noise characteristics and the density function adaptation of the sampling. In [49], a variant of Lloyd's method is defined by considering weighted centres. In [16], the Voronoi tessellation is built according to a density function defined on the input domain. In [8], a variational approach is defined by combining the centroidal Voronoi tessellation with the capacity-constrained one. In [51], bubbles are placed on the domain and are governed by inter-bubble forces; when a stable configuration is reached through a dynamic simulation, the tessellation is formed by connecting the centres of adjacent bubbles. In [9], surface sampling and reconstruction are computed through iterative centroidal Voronoi tessellation, based on a local approximation of the surface with a best-fitting planes. The generation of blue-noise sampling through Wang tessellation [28] is achieved with a recursive approach, which adaptively splits the tiles and relaxes the tessellation to match the point set.

Optimal transport [27] is applied to generate high dimensional sampling for the computation of the Monte Carlo integration of a generic function. In [12], the capacity-constrained Voronoi tessellation is formulated as a continuous minimisation problem based on optimal transport and is applied to generate a blue-noise sampling without local artifacts. The optimal transport problem is progressively solved on a sequence of discrete measures [34] that converge to the optimal solution. In [40], the sliced optimal transport projects and solves the problem onto repeated 1D dimensions; in particular, the distance to be minimised is defined as the integral over the slice directions between the projections of the input points on the selected direction and the orthogonal projection of the density function. In [44], a multi-class sampling is computed as a constrained

barycentre of probability measures.

3. Enhanced kernel-based sampling

We introduce the proposed variants of the kernel-based sampling (Sect. 3.1), the experimental results (Sect. 3.2), a comparison between samples' position (Sect. 3.3), and between different reconstruction methods (Sect. 3.4).

3.1. Enhanced kernel-based sampling

The kernel-based sampling [64] approximates an input signal as a linear combination of Gaussian kernel functions, whose centres (i.e., the samples) are computed through the minimisation of the energy functional. We propose a novel method, the $(\boldsymbol{\sigma}, \boldsymbol{\alpha})$ *kernel-based sampling*, that approximates an input signal $C(\mathbf{x})$ on $\mathbb{R}^d$ as a linear combination

$$\begin{cases} C_{\text{recon}}(\mathbf{x}, \boldsymbol{\mu}, \boldsymbol{\alpha}, \boldsymbol{\sigma}) = k \sum_{j=1}^{n} \alpha_j G(\mathbf{x}, \boldsymbol{\mu}_j, \sigma_j), \\ G(\mathbf{x}, \boldsymbol{\mu}_j, \sigma_j) := \frac{1}{(\sqrt{2\pi}\sigma_j)^d} \exp(-\frac{\|\mathbf{x}-\boldsymbol{\mu}_j\|_2^2}{2\sigma_j^2}). \end{cases} \tag{1}$$

of functions $G(\mathbf{x}, \boldsymbol{\mu}_j, \sigma_j)$, with centres $\boldsymbol{\mu} = \{\boldsymbol{\mu}_j\}_{j=1}^n$, weights $\boldsymbol{\alpha} := (\alpha_j)_{j=1}^n$, widths $\boldsymbol{\sigma} := (\sigma_j)_{j=1}^n$, and $k = \int_\Omega C(\mathbf{x}) d\mathbf{s}/n$ as a constant term. Then, the variables $(\boldsymbol{\mu}, \boldsymbol{\sigma}, \boldsymbol{\alpha})$ are computed by minimising the energy functional

$$E(\boldsymbol{\mu}, \boldsymbol{\alpha}, \boldsymbol{\sigma}) = \int_\Omega |C(\mathbf{x}) - C_{\text{recon}}(\mathbf{x}, \boldsymbol{\mu}, \boldsymbol{\alpha}, \boldsymbol{\sigma})|^2 d\mathbf{s}, \tag{2}$$

In the discrete case, the input signal is known at a set of points $\mathcal{P} = \{\mathbf{x}_i\}_{i=1}^m$ and the integral in Eq. (2) is discretized as a finite sum over the input points. Introducing the integrand term $\bar{S}^2(\mathbf{x}, \boldsymbol{\mu}, \boldsymbol{\sigma}, \boldsymbol{\alpha})$ of the energy functional (2), the minima of $E(\cdot)$ are computed as the roots of its partial derivatives with respect to $\boldsymbol{\mu}$, $\boldsymbol{\sigma}$, $\boldsymbol{\alpha}$, i.e.,

$$\frac{\partial E(\boldsymbol{\mu}, \boldsymbol{\alpha}, \boldsymbol{\sigma})}{\partial \boldsymbol{\mu}_i} = \frac{-2k}{(\sqrt{2\pi})^d \sigma_i^{d+2}} \alpha_i \int_\Omega \bar{S}(\mathbf{x}, \boldsymbol{\mu}, \boldsymbol{\sigma}, \boldsymbol{\alpha}) \exp(-\frac{\|\mathbf{x} - \boldsymbol{\mu}_i\|_2^2}{2\sigma_i^2})(\mathbf{x} - \boldsymbol{\mu}_i) d\mathbf{s};$$

$$\frac{\partial E(\boldsymbol{\mu}, \boldsymbol{\alpha}, \boldsymbol{\sigma})}{\partial \alpha_i} = \frac{-2k}{(\sqrt{2\pi}\sigma_i)^d} \int_\Omega \bar{S}(\mathbf{x}, \boldsymbol{\mu}, \boldsymbol{\sigma}, \boldsymbol{\alpha}) \exp(-\frac{\|\mathbf{x} - \boldsymbol{\mu}_i\|_2^2}{2\sigma_i^2}) d\mathbf{s};$$

$$\frac{\partial E(\boldsymbol{\mu}, \boldsymbol{\alpha}, \boldsymbol{\sigma})}{\partial \sigma_i} = \frac{-2k}{(\sqrt{2\pi}\sigma_i)^d} \int_\Omega \bar{S}(\mathbf{x}, \boldsymbol{\mu}, \boldsymbol{\sigma}, \boldsymbol{\alpha}) \exp(-\frac{\|\mathbf{x} - \boldsymbol{\mu}_i\|_2^2}{2\sigma_i^2}) \left[\frac{-d}{\sigma_i} + \frac{\|\mathbf{x} - \boldsymbol{\mu}_i\|_2^2}{\sigma_i^3}\right] d\mathbf{s}.$$

Further details on the energy functionals and derivatives of the kernel-based sampling are discussed in the Appendix.

We now focus on the properties of the $(\boldsymbol{\sigma}, \boldsymbol{\alpha})$ kernel-based sampling, the evaluation of the energy functional, and its computational cost. The $(\boldsymbol{\sigma}, \boldsymbol{\alpha})$ kernel-based sampling involves $n(d+2)$ variables, i.e., nd variables for the d coordinates of the n samples $\boldsymbol{\mu}$, n variables for the coefficients $\boldsymbol{\alpha}$, and n variables for the Gaussian width $\boldsymbol{\sigma}$. We specialise this method to four variants, by freezing a subset of the free variables in the approximating function (1):

- $(\boldsymbol{\mu})$ *kernel-based sampling* (or $(\boldsymbol{\mu})$-method): we select $\alpha_j = 1$, $\sigma_j = \sigma$, $\forall j$, and σ is a constant, thus reducing to the original kernel-based method [64], with nd variables;

- $(\boldsymbol{\sigma})$ *kernel-based sampling* (or $(\boldsymbol{\sigma})$-method): we select $\alpha_j = 1$, $\forall j$ and the corresponding energy functional involves $n(d+1)$ variables;

- $(\boldsymbol{\alpha})$ *kernel-based sampling* (or $(\boldsymbol{\alpha})$-method): we select $\sigma_j = \sigma$, $\forall j$, and σ is constant. The corresponding energy functional involves $n(d+1)$ variables;

- (σ) *kernel-based sampling* (or (σ)-method): we select $\alpha_j = 1$, $\sigma_j = \sigma$, $\forall j$, and σ is a variable. The corresponding energy functional involves $nd+1$ variables.

Signal reconstruction and error metrics. Once the samples have been computed, we reconstruct the input signal at any point $\mathbf{y}$ as a linear combination of the kernel functions, by computing $C_{\text{recon}}(\mathbf{y}) := C_{\text{recon}}(\mathbf{y}, \boldsymbol{\mu}, \boldsymbol{\alpha}, \boldsymbol{\sigma})$ in Eq. (1). Then, we evaluate the reconstruction accuracy as the difference $|C_{\text{recon}}(\mathbf{x}_i) - C(\mathbf{x}_i)|$ between the input and the reconstructed signals at $\mathbf{x}_i$, $\forall i$. Given m input points, we evaluate the *normalised cross correlation* (NCC)

$$\text{NCC} = \frac{\sum_{i=1}^{m} [C_{\text{recon}}(\mathbf{x}_i) - \bar{C}_{\text{recon}}][C(\mathbf{x}_i) - \bar{C}]}{\left[\sum_{i=1}^{m} [C_{\text{recon}}(\mathbf{x}_i) - \bar{C}_{\text{recon}}]^2\right]^{1/2} \left[\sum_{i=1}^{m} [C(\mathbf{x}_i) - \bar{C}]^2\right]^{1/2}}, \tag{3}$$

where $\bar{C}_{recon}$ and $\bar{C}$ are the average values of the reconstructed and input signal respectively, and the *normalised root mean square error* (NRMSE)

$$\text{NRMSE} = \left[\frac{\sum_{i=1}^{m} [C_{\text{recon}}(\mathbf{x}_i) - C(\mathbf{x}_i)]^2}{\sum_{i=1}^{m} [C(\mathbf{x}_i)]^2} \right]^{1/2}.$$

Then, the *cumulative error* computes the aggregated error at each input point $\mathbf{x}_i$

$$E(\mathbf{x}_i) = \left[\sum_{j=1}^{i} |C_{\text{recon}}(\mathbf{x}_j) - C(\mathbf{x}_j)|^2 \right]^{1/2}, \tag{4}$$

and the P_k*-percentile* is defined as the percentage of input points whose reconstruction error is lower than k, i.e.,

$$P_k = \frac{\#\{i : |C_{\text{recon}}(\mathbf{x}_i) - C(\mathbf{x}_i)| < k\}}{m}.$$

In the paper examples, we also visualise the error as the difference between the input and the approximated images, where white corresponds to a null error and black represents the maximum error equal to one (c.f., Fig. 2).

Numerical solver and computational cost. The minimum of the discrete energy functional is computed through the iterative optimisation method L-BFGS (*Limited-memory Broyden, Fletcher, Goldfarb, Shanno*) [65], which finds the roots of the derivative of the energy functional. We briefly recall that L-BFGS is an optimisation algorithm in the family of quasi-Newton methods that approximates the Broyden-Fletcher-Goldfarb-Shanno algorithm (BFGS) using a limited amount of computer memory. Analogously to BFGS, the L-BFGS solver estimates the inverse Hessian matrix for the minimum search in the variable space; however, the L-BFGS method represents the approximation implicitly through a few vectors, thus involving a limited memory requirement. At each iteration, a small history of the past updates of the position $(\boldsymbol{\mu}, \boldsymbol{\alpha}, \boldsymbol{\sigma})$ and of the gradient of the energy functional $E(\cdot)$ in Eq. (2) is used to identify the direction of steepest descent and to implicitly perform operations requiring vector products with the inverse Hessian matrix. For the L-BFGS method, the memory storage is $\mathcal{O}(u^2)$ and the computational cost is $\mathcal{O}(uv)$ at each iteration, where u is

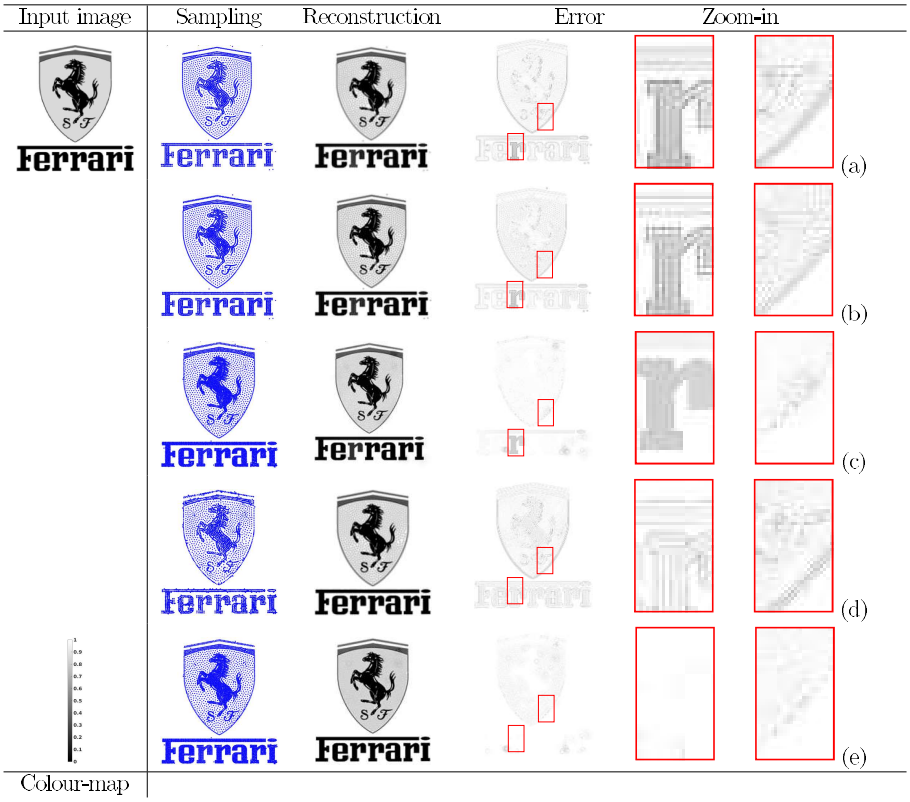


Figure 2: Variants comparison: (a), ($\boldsymbol{\mu}$)-method; (b), (σ)-method; (c), ($\boldsymbol{\sigma}$)-method; (d), ($\boldsymbol{\alpha}$)-method; (e), ($\boldsymbol{\sigma}, \boldsymbol{\alpha}$)-method, with 5K samples. First column shows the input image (256×256), and the colour-map.

the total number of variables, and v is the number of steps stored in memory. The computational cost of the kernel-based sampling is $\mathcal{O}(u + m)$, where u is the number of variables and m is the number of input points, and the memory storage is $\mathcal{O}(u^2)$.

3.2. Experimental results

We compare the results of the proposed variants of the kernel-based sampling (Fig. 2). The main geometric features of the input image are preserved by all the variants; in particular, the tail and the mane of the horse, or the "Ferrari" label. The reconstruction of the input image with the original kernel-based sampling presents some defects, such as a blurring on the letter "R", noise around the

letter "I", and a dot-like effect on the grey background of the logo. The (σ)-method has a similar result compared to the ($\boldsymbol{\mu}$)-method, with mainly the same defects. The ($\boldsymbol{\sigma}$)- and ($\boldsymbol{\alpha}$)-methods provide some improvements to the previous results. In the ($\boldsymbol{\sigma}$)-method, the dot-like effect on the grey background is barely visible. The characters are well reconstructed, apart from the letter "R", which is still blurred. The noise around the letter "I" is not present.

A different reconstruction accuracy of the "R" letters (i.e., the first one is more blurred than the second and the third ones) depends on the overall number of samples and on the number of samples that belong to this area of the image; in (a), 190 samples are placed in the first "R" letter, 273 and 282 samples are placed in the second and third "R" letter respectively, thus leading to a different approximation and reconstruction of this area. The number of samples for each "R" letter depends on two factors: the samples' initialisation and the iterative optimisation of the energy functional. Since the optimisation of the energy functional is deterministic and the samples' initialisation is stochastic [64], a different initialisation of the method leads to a larger or smaller number of samples for each "R" letter, and consequently, to a different reconstruction of the corresponding area of the image.

The ($\boldsymbol{\alpha}$)-method is affected by a dot-like effect on the grey background; furthermore, the "Ferrari" label looks more jagged and irregular. The reconstruction accuracy of the ($\boldsymbol{\sigma}, \boldsymbol{\alpha}$)-method is very good; in fact, there is no blurring or dot-like effects, the background is very uniform, and the grey distribution of the original image is preserved. Furthermore, all the letters of the "Ferrari" label are perfectly reconstructed. Considering the error distribution between the input and the reconstructed images, the ($\boldsymbol{\mu}$)-method has a visible error on most of the characters of "Ferrari"; in particular, on the letter "R". Also, the error is higher on the external boundary of the shapes. The error of the (σ)-method is analogous to the error of the ($\boldsymbol{\mu}$)-method, while it is significantly reduced for the ($\boldsymbol{\sigma}$)-method (e.g., in the logo background) and for the ($\boldsymbol{\alpha}$)-method (e.g., on the letter "R"). The error is barely visible in the ($\boldsymbol{\sigma}, \boldsymbol{\alpha}$)-method, including the "Ferrari" label.

Table 1: With reference to Fig. 2, we report the approximation accuracy of the five kernel-based sampling variants, with best results in bold.

Method	$(\boldsymbol{\mu})$	(σ)	$(\boldsymbol{\sigma})$	$(\boldsymbol{\alpha})$	$(\boldsymbol{\sigma}, \boldsymbol{\alpha})$
Objective function	149.9	112.0	29.8	73.0	**6.65**
NCC	0.988	0.990	0.996	0.992	**0.998**
NRMSE	0.048	0.044	0.025	0.043	**0.019**
$P_{0.05}$	86.5%	86.5%	96.4%	86.3%	**96.9%**
$P_{0.10}$	94.2%	95.3%	98.8%	95.2%	**99.5%**

We conclude that the $(\boldsymbol{\sigma})$-method provides better results than $(\boldsymbol{\alpha})$-method, particularly in the background reconstruction; indeed, the use of the $\boldsymbol{\sigma}$ variables improves the quality of the approximation of the input image with respect to the $\boldsymbol{\alpha}$ variables, despite the number of variables remains unchanged. The $(\boldsymbol{\sigma}, \boldsymbol{\alpha})$-method gives the best results, with an excellent sampling, approximation, and reconstruction of the input image.

Approximation accuracy. In Table 1, we present a comparison of the five kernel-based sampling methods, in terms of the error metrics described in Sect. 3.1. First of all, each variant of the kernel-based sampling further reduces the numerical value of the objective function with respect to the $(\boldsymbol{\mu})$-method. In fact, the minimum of the energy functional is computed on a larger trust region. The $(\boldsymbol{\mu})$-method has an objective function value of 150, a $P_{0.10}$ value of 94.2% and a NCC value of 0.988. The (σ)-method has an objective function value of 112, it improves the $P_{0.10}$ value to 95.3%, and the NCC value to 0.990. The $(\boldsymbol{\sigma})$-method has an objective function value of 29.8; it has very good $P_{0.05}$ and $P_{0.10}$ values, 96.4% and 98.8% respectively. The $(\boldsymbol{\alpha})$-method has an objective function value of 73; the NCC, the $P_{0.05}$, and the $P_{0.10}$ values are similar to the ones of the (σ)-method. Finally, the $(\boldsymbol{\sigma}, \boldsymbol{\alpha})$-method significantly improves all the metrics for the error evaluation, since it has an objective function value of 6.65, the $P_{0.10}$ value is 99.5%, and the NCC value is 0.998.

The (σ)-method has better results than the original method, with the addition of one variable only. The $(\boldsymbol{\sigma})$- and $(\boldsymbol{\alpha})$-methods have worse results than the $(\boldsymbol{\sigma}, \boldsymbol{\alpha})$-method, and the $(\boldsymbol{\sigma})$-method has better results than the $(\boldsymbol{\alpha})$-method,

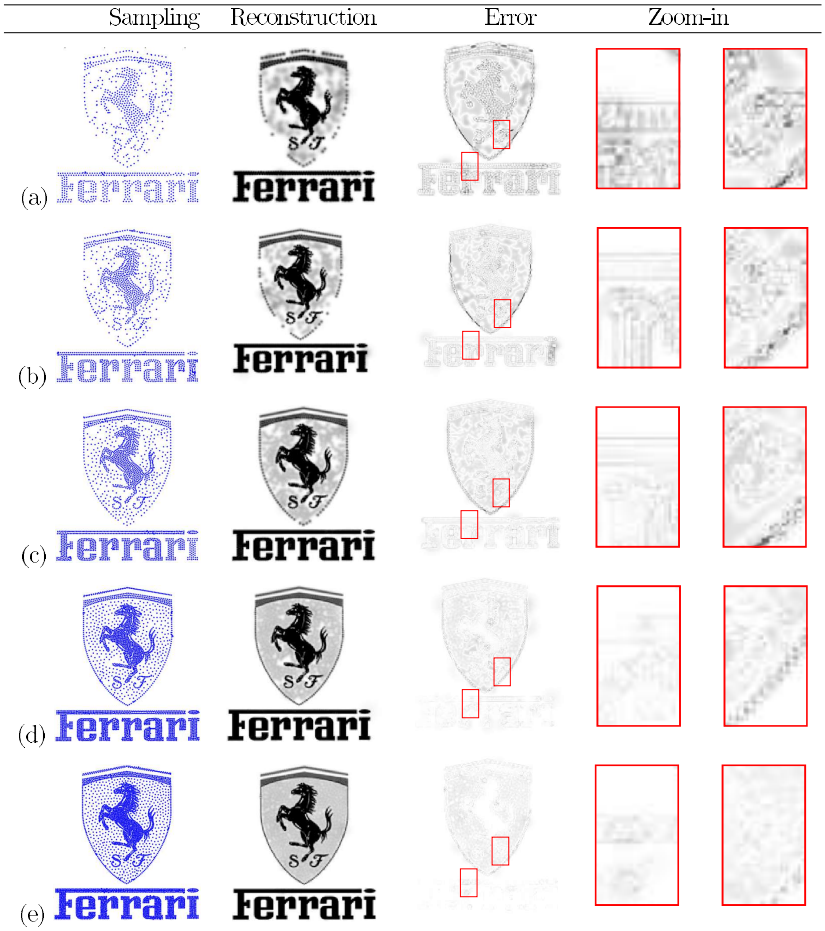


Figure 3: Accuracy of the $(\boldsymbol{\sigma}, \boldsymbol{\alpha})$-method, with a different number of samples: (a) $n = 1K$; (b) $n = 1.5K$; (c) $n = 2K$; (d) $n = 3K$; (e) $n = 4K$.

despite the same number of variables. Finally, the $(\boldsymbol{\sigma}, \boldsymbol{\alpha})$-method significantly improves the $(\boldsymbol{\mu})$-method under all the metrics.

Approximation accuracy with respect to the number of samples. Since the $(\boldsymbol{\sigma}, \boldsymbol{\alpha})$-method has the highest approximation quality, we further analyse its accuracy with respect to the number of samples. Selecting a different number of samples (i.e., from 1K to 4K), the blurring and the dot-like effects are more accentuated with a lower number of samples; however, the overall features (e.g., the horse shape, the "Ferrari" label), are well reconstructed. The reconstruction error de-

Table 2: With reference to Fig. 3, we report the error metrics for the $(\boldsymbol{\sigma}, \boldsymbol{\alpha})$-method with a different number of samples; best results are in bold.

Samples	5K	4K	3K	2K	1.5K	1K
Objective function	**6.65**	17.5	50.1	123.2	200.2	350.4
NCC	**0.998**	0.997	0.995	0.989	0.982	0.971
NRMSE	**0.019**	0.025	0.033	0.048	0.064	0.088
$P_{0.05}$	**96.9%**	92.8%	89.4%	83.8%	80.6%	77.4%
$P_{0.10}$	**99.5%**	99.4%	98.3%	94.6%	90.3%	84.7%

creases as the number of samples increases; especially on the letter "R" and on the background of the logo (Fig. 3).

Comparing the error metrics (Table 2), the $(\boldsymbol{\sigma}, \boldsymbol{\alpha})$-method is very accurate, even when we use only 1K samples (i.e., the 1.5% of the input pixels). In fact, 77% of the points have a reconstruction error lower than 0.05, and 85% of the points have an error lower than 0.10. We underline that the accuracy of the $(\boldsymbol{\sigma}, \boldsymbol{\alpha})$-method with 2K samples is comparable to the $(\boldsymbol{\mu})$-method with 5K samples (Table 1).

Computation time. Table 3 shows the variation of the execution time and the number of iterations of the L-BFGS algorithm to converge, with respect to the original kernel-based sampling (i.e., the $(\boldsymbol{\mu})$-method). Tests have been performed on a desktop with 3,1 GHz Dual-Core Intel Core i7, and 16GB RAM. The $(\boldsymbol{\mu})$-method takes 67 iterations and 24 seconds to converge to the solution. The (σ)-method increases the execution time by 2.3 times and the number of iterations to 149. The $(\boldsymbol{\sigma})$-method has a very high execution time, due to the much larger number of variables, and to the slow convergence of the algorithm (357 iterations). The $(\boldsymbol{\alpha})$-method takes only 83 iterations, thus resulting very close to the original kernel-based sampling, but its execution time is increased by 1.9 times. The $(\boldsymbol{\sigma}, \boldsymbol{\alpha})$-method increases the execution time by 9.1 times, and it takes 674 iterations to converge to the solution.

Fig. 4 shows the value of the objective function (y-axis) with respect to the number of iterations (x-axis) for the five methods. The $(\boldsymbol{\sigma})$- and the $(\boldsymbol{\sigma}, \boldsymbol{\alpha})$-methods have a slower convergence to zero with respect to the other variants

Table 3: With reference to Fig. 2, we report the increment of the execution time and the number of iterations for the five variants, with respect to the execution time $T = 24$s (in seconds s) of the $(\boldsymbol{\mu})$-method.

Method	$(\boldsymbol{\mu})$	(σ)	$(\boldsymbol{\sigma})$	$(\boldsymbol{\alpha})$	$(\boldsymbol{\sigma}, \boldsymbol{\alpha})$
Execution Time [s]	T	$2.3T$	$8.4T$	$1.9T$	$9.1T$
Iterations	67	149	357	83	674

Table 4: Memory allocation of the kernel-based sampling, when using $n = 5K$ samples; each variable is stored as a double precision number.

Method	$(\boldsymbol{\mu})$	(σ)	$(\boldsymbol{\sigma})$	$(\boldsymbol{\alpha})$	$(\boldsymbol{\sigma}, \boldsymbol{\alpha})$
Memory allocation [GB]	0.74	0.74	1.68	1.68	2.98

of the kernel-based sampling. According to Table 4 and selecting $n = 5K$ samples, the allocated memory increases of about four times, when passing from the $(\boldsymbol{\mu})$- to the $(\boldsymbol{\sigma}, \boldsymbol{\alpha})$-method. Table 5 shows the execution time and the number of iterations of the $(\boldsymbol{\sigma}, \boldsymbol{\alpha})$-method, when varying the number of samples. Decreasing the number of samples the number of iterations increases, as the solver converges more slowly.

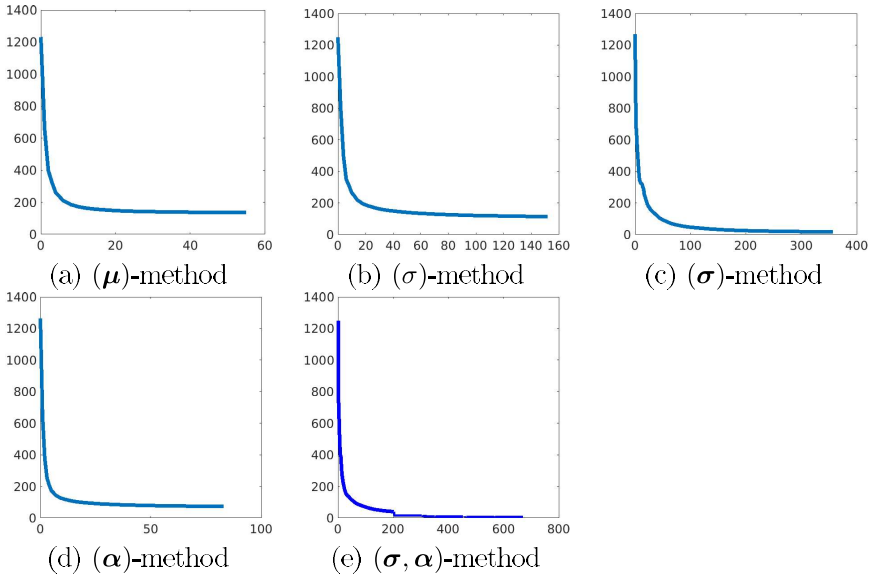
3.3. Comparison between samplings

We compare the $(\boldsymbol{\mu})$ and $(\boldsymbol{\sigma}, \boldsymbol{\alpha})$ kernel-based samplings in terms of their Hausdorff distance and the approximation accuracy of the reconstructed signals. We recall that the *symmetric Hausdorff distance* of two point sets $\mathcal{X}, \mathcal{Y}$ is defined as $d(\mathcal{X}, \mathcal{Y}) := \max\{d_{\mathcal{X}}(\mathcal{Y}), d_{\mathcal{Y}}(\mathcal{X})\}$, with $d_{\mathcal{X}}(\mathcal{Y}) := \max_{\mathbf{x}\in\mathcal{X}}\{\min_{\mathbf{y}\in\mathcal{Y}}\{\|\mathbf{x} - \mathbf{y}\|_2\}\}$ *one-side Hausdorff distance*.

In Fig. 5, we show the NCC error of the $(\boldsymbol{\mu})$- and the $(\boldsymbol{\sigma}, \boldsymbol{\alpha})$-methods, and the Hausdorff distance (y-axis) between the sampling of the two methods, when varying the number of samples from 500 to 5K (x-axis). The variation of the values of the Hausdorff distance shows that the samplings of the two methods remain different as we increase the number of samples. The NCC trend shows that the $(\boldsymbol{\sigma}, \boldsymbol{\alpha})$ kernel-based sampling remains more accurate than the standard sampling. In Table 6, we report the Hausdorff distance between the sampling of the four variants (b-e) and of the $(\boldsymbol{\mu})$-method (i.e., the kernel-based sampling [64]); the sampling of the $(\boldsymbol{\sigma})$-method is the closest one to the original kernel-based sampling, while the sampling of the (σ)-method is the farthest.

Table 5: With reference to Fig. 3, we report the reduction of the execution time and the number of iterations for the $(\boldsymbol{\sigma}, \boldsymbol{\alpha})$-method with a different number of samples, with respect to the execution time $T = 216$s with 5K samples.

Samples	5K	4K	3K	2K	1.5K	1K
Execution time [s]	T	T	$0.6T$	$0.4T$	$0.4T$	$0.2T$
Iterations	657	896	785	929	891	1052



(a) $(\boldsymbol{\mu})$-method (b) (σ)-method (c) $(\boldsymbol{\sigma})$-method

(d) $(\boldsymbol{\alpha})$-method (e) $(\boldsymbol{\sigma}, \boldsymbol{\alpha})$-method

Figure 4: With reference to Fig. 2, we report the convergence of the iterative minimisation, i terms of the objective function value (y-axis) with respect to the number of iterations (x-axis).

We also compare our approach with state-of-the-art sampling methods on an intensity-increasing image, counting the number of samples for each quarter of the ramp image (Fig. 6). Our method achieves results analogous to [64] and is comparable with state-of-the-art methods, in terms of sampling. Furthermore, it better reconstructs the input signal with respect to [64] (Fig. 7), improving the preservation of grey-scale values and reducing the scattering effect. Considering the quantitative metrics of the reconstructed images, the NCC value is 0.987 and the NRMSE value is 0.084 for our method, while these values are 0.946 and 0.151 respectively in [64].

Table 6: With reference to Fig. 2, we report the Hausdorff distance normalised with the diagonal of the image, between the four variants (b-e) and the original kernel-based sampling (i.e., the $(\boldsymbol{\mu})$-method).

Method	(σ)	$(\boldsymbol{\sigma})$	$(\boldsymbol{\alpha})$	$(\boldsymbol{\sigma}, \boldsymbol{\alpha})$
Hausdorff distance	0.0157	0.0104	0.0117	0.0105

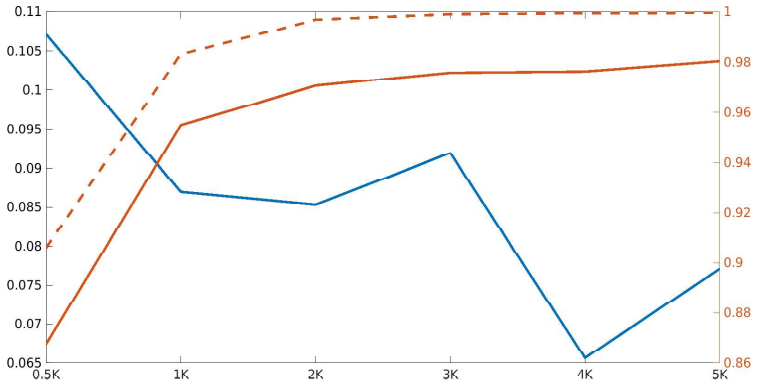


Figure 5: NCC error (y-axis, right): $(\boldsymbol{\mu})$- (red continuous line) and $(\boldsymbol{\sigma}, \boldsymbol{\alpha})$-(red dashed line) kernel-based samplings; Hausdorff distance (y-axis, left) of the corresponding samples (blue line), with respect to their number (x-axis). Image: Dante, 256×256.

3.4. Kernel-based sampling and least-squares approximation

Once the samples have been computed, we reconstruct the input signal as

$$C_{LS}(\mathbf{x}) := \sum_{j=1}^{n} \gamma_j G(\mathbf{x}, \boldsymbol{\mu}_j, \sigma_j) =_{\text{Eq.(1)}} C_{\text{recon}}(\mathbf{x}, \boldsymbol{\mu}, \boldsymbol{\gamma}, \boldsymbol{\sigma}), \tag{5}$$

where $\boldsymbol{\gamma}$ is the $n \times 1$ coefficients' vector that solve the least-squares linear system $\mathbf{G}\boldsymbol{\gamma} = \mathbf{C}$. Here, $\boldsymbol{G}$ is the $m \times n$ Gram matrix associated with the Gaussian kernel, evaluated at the m points $(\mathbf{x}_i)_{i=1}^m$, and at the n samples $(\boldsymbol{\mu}_j)_{j=1}^n$, $\mathbf{C} := (C(\mathbf{x}_i))_{i=1}^m$ is the $m \times 1$ vector associated with the input signal. According to Table 7, the least-squares reconstruction has a better accuracy than the standard kernel-based reconstruction through Eq. (1) when using the $(\boldsymbol{\mu})$-method, while the two reconstruction approaches have the same accuracy when the $(\boldsymbol{\sigma}, \boldsymbol{\alpha})$-method is used.

4. Sampling initialisation and optimisation

We introduce two improvements of the kernel-based sampling in terms of (i) *sampling initialisation* (Sect. 4.1), which allows us to process high resolution

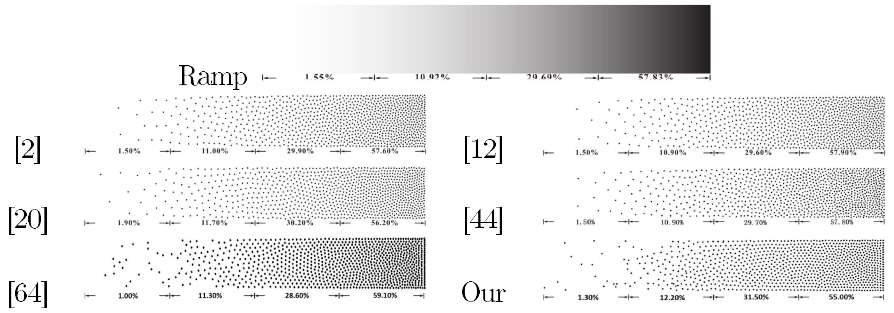


Figure 6: Ramp image (192×75) and sampling (1K samples) comparison among state-of-the-art methods; the percentages of each quarter show the number of samples in that portion of the grid, compared with the ink density of the input image.

Figure 7: Comparison of the reconstructed ramp image, between (a) [64], and (b) ours.

images without memory overload, and (ii) *sampling optimisation* (Sect. 4.2), which selects the number of samples with an iterative algorithm that places the samples according to the local approximation accuracy. These results can be applied to any method in Sect. 3.1; in our experiments, we focus on the $(\boldsymbol{\sigma}, \boldsymbol{\alpha})$ kernel-based sampling.

4.1. Sampling initialisation

For the sampling initialisation, our goal is to increase the number of samples and the resolution of the input image, overcoming memory limits that previous methods might encounter when computing/allocating the partial derivatives and evaluating the minimum value of the energy functional (Table 4). To this end, we split the input image into a set of images, and we work independently on each of them. More precisely, given a 2D image Ω, we split it into m 2D images $(\Omega_i)_{i=1}^m$, and the number of samples of each sub-image is defined according to its grey intensity $C_{\Omega_i} = \int_{\Omega_i} C(\mathbf{x}) d\mathbf{s}$ as $n_{\Omega_i} = (C_{\Omega_i}/C_\Omega) n_\Omega$, where n_Ω is the overall number of samples. In this way, we avoid that too many samples are placed in those regions where the white is predominant.

Table 7: NCC values for the reconstruction associated with the $(\boldsymbol{\mu})$-, and $(\boldsymbol{\sigma}, \boldsymbol{\alpha})$- methods and applied to Pinocchio 256×256.

Num. samples	**Reconstr. method**	$(\boldsymbol{\mu})$	$(\boldsymbol{\sigma}, \boldsymbol{\alpha})$
2K	Gaussian kernels (Eq. (1))	0.6612	0.7944
	C_{LS} (Eq. (5))	0.7962	0.7944
5K	Gaussian kernels (Eq. (1))	0.8805	0.9637
	C_{LS} (Eq. (5))	0.9372	0.9637

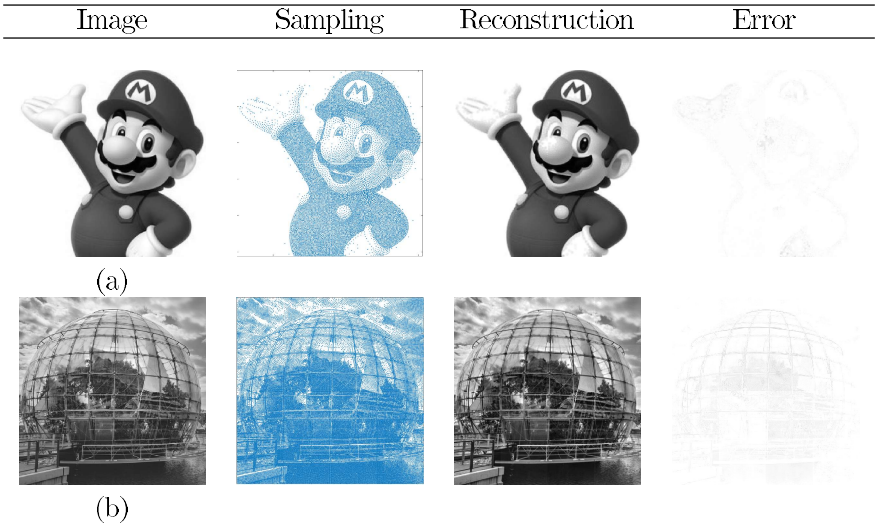


Figure 8: Sampling initialisation on (a) Supermario (256×256 input points, 16K samples, $m = 4$, (b) Biosphere (512×512 input points, 50K samples, $m = 16$).

To further improve the reconstructed image, we add a small overlay to the sub-images to be sampled: Supermario (Fig. 8(a), 256×256) is divided into four sub-images of 132×132, with an overlap of 132×8 for each couple of adjacent sub-images, and has been sampled with an overall number of $n = 16K$ samples. In a similar way, the Biosphere image (512×512) is divided into 16 sub-images of 136×136 resolution, with an overlap of 136×16 for each couple of adjacent sub-images, with $n = 50K$ samples. Then, the reconstructed image in the overlapped area is computed as the average between the two reconstructed sub-images.

For both examples, the sampling and the reconstructed image preserve the main features of the two subjects. In Supermario, the samples are well localised

Table 8: With reference to Fig. 8, we report the approximation accuracy of the sampling optimisation.

Image	Supermario	Biosphere
NCC	0.9996	0.985
NRMSE	0.0125	0.0884
$P_{0.05}$	99.3%	78.4%
$P_{0.10}$	99.9%	95.3%

in dark areas (e.g., the hat, the body, the moustaches), and all the features are well reconstructed (e.g., the letter "M"). The reconstruction error is very low and uniform. The Biosphere is more difficult to be sampled and reconstructed, due to its high resolution, the presence of complex geometries and their overlaps, e.g., the sphere, clouds, plants, and shadows on the sea. The approximation error is lower on the sphere structure, which is well reconstructed, and generally higher on the clouds and on the sea shadows. According to Table 8, the approximation accuracy is very good for both the examples, due to a high number of selected samples. In particular, the NCC value of Supermario is 0.9996, compared to the NCC value of 0.9954 when using 5K samples.

4.2. Sampling optimisation

The sampling optimization computes the number of samples and their location according to a target approximation accuracy, without pre-defining the number of samples. The samples are added incrementally through an iterative approach; at each iteration, the kernel-based sampling is applied to the reconstruction error image in order to add new samples where the approximation error is higher, thus increasing the approximation accuracy and reducing the overall computation time.

Given the Ferrari input image (Fig. 2, first column), we apply the kernel-based sampling by adding 500 samples at each iteration. Fig. 9(a) shows the sampling at the first iteration; the image reconstruction is very rough, due to the low number of samples, and the "Ferrari" label and the horse logo have a high reconstruction error. After 10 iterations (5K samples), the geometries and the relevant features of the logo are preserved (Fig. 9(d)). According to Table 9,

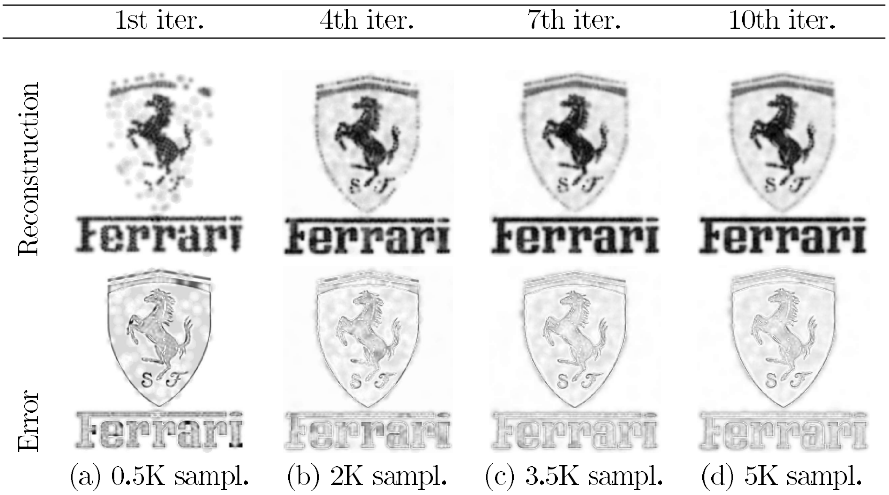


Figure 9: Image reconstruction and error for the sampling optimisation: (a) first iteration, 500 samples; (b) fourth iteration, 2K samples; (c) seventh iteration, 3.5K samples; (d) tenth iteration, 5K samples.

Table 9: With reference to Fig. 9, we report the approximation accuracy of the adaptive kernel-based sampling.

Samples	500	2K	3.5K	5K
NCC	0.929	0.957	0.962	0.964
NRMSE	0.148	0.109	0.101	0.098
$P_{0.05}$	73.4%	74.4%	74.4%	74.5%
$P_{0.10}$	79.5%	83.6%	84.3%	84.9%

increasing the number of samples, the approximation accuracy improves; for instance, the NCC value increases from 0.929 with 500 samples, to 0.964 with 5K samples.

5. Kernel-based sampling of arbitrary data

To show the generality of the proposed $(\boldsymbol{\sigma}, \boldsymbol{\alpha})$ kernel-based sampling, we apply this method to 3D data (Sect. 5.1), simulation data on unstructured grids (Sect. 5.2), and to vector-valued functions (Sect. 5.3). Finally (Sect. 5.4), we discuss the main limitations of the proposed approach.

5.1. Kernel-based sampling of 3D data

Fig. 10 and Table 10 show the results of the sampling and reconstruction of a volumetric MR image, with size $128 \times 128 \times 24$; the initialisation (Sect. 4.1)

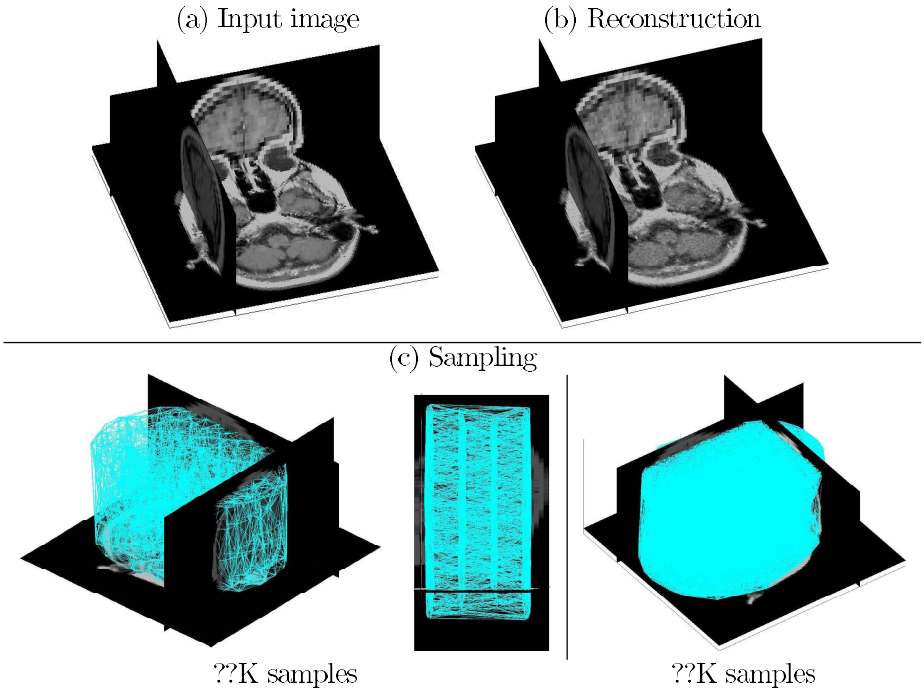


Figure 10: (a) input MR image ($128 \times 128 \times 24$), (b) reconstruction, and sampling with (c, left) ??K samples (2 views) and (c, right) ??K samples.

with 50K samples is applied to 8 sub-images of size $128 \times 128 \times 3$. The sampling covers the entire brain, and the reconstructed image accurately approximates the main features of the input data. In this test, the samples are denser where the input signal is higher (e.g., white parts of the image), and our method improves the results of [64], as the NCC value increases from 0.82 to 0.91, and the $P_{0.10}$ metrics increases from 0.1 to 0.4. Due to a larger number (i.e., 250K) of variables, the execution time (2700s) increases with respect to [64] (i.e., 150K variables).

5.2. Kernel-based sampling of unstructured data

We apply the $(\boldsymbol{\sigma}, \boldsymbol{\alpha})$-method to an input signal defined on an unstructured grid, with the aim of improving the sampling density where the signal is more significant. For instance, if the PDE models a mechanical stress then we expect to place a higher number of samples in those regions of a 2D or 3D domain

Table 10: With respect to Fig. 10, we report the quantitative metrics and the execution time of our method and the kernel-based sampling [64].

Method	[64]	**Ours**
NCC	0.82	0.91
NRMSE	1.00	0.83
$P_{0.05}$	< 0.1	0.18
$P_{0.10}$	< 0.1	0.59
Execution time[s]	1020	2700

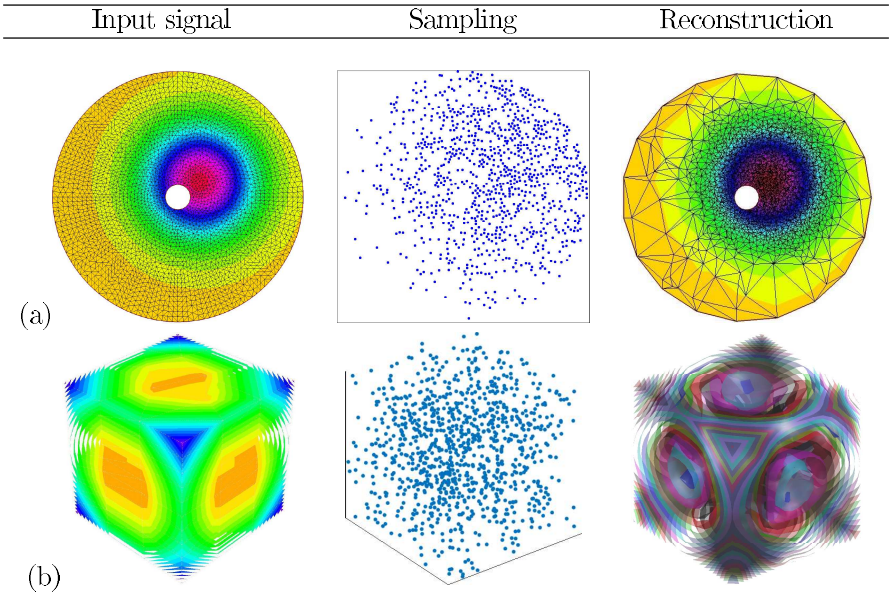


Figure 11: Kernel-based sampling on unstructured grid: (a) 2D annulus (20K input points, 2K samples), (b) 3D cube (5K input points, 500 samples).

where the simulated stress is higher.

As input, we consider the solution of the heat equation on an annulus (Fig. 11(a), first column), which is discretized as an unstructured grid of 20K points. The $(\boldsymbol{\sigma}, \boldsymbol{\alpha})$-method places more samples in those regions of the annulus where the intensity of the heat is higher. Recomputing the heat distribution (third column) by solving the heat equation on the mesh whose vertices are the computed samples, the solution accuracy is preserved in those regions where we have higher heat values, due to the density distribution of the samples. Fig. 11(b, first column) shows the solution of the Laplace equation on a cube,

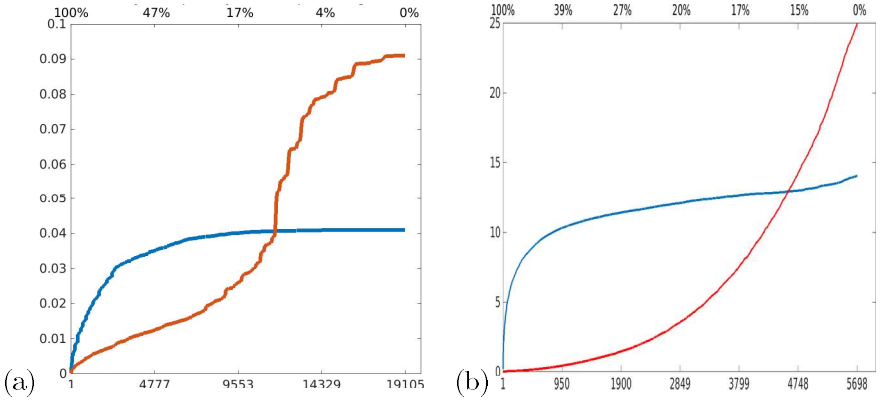


Figure 12: With reference to Fig. 11, we report the cumulative error (y-axis), with respect to the input points (x-axis, bottom), and to the intensity of the signal (x-axis, top), for both the 2D (a) and the 3D (b) case. The red line shows the error of the solution (ii), while the blue line shows the error of the solution (iii).

computed on an unstructured 3D grid with 5K points. Then, 500 samples are placed with the kernel-based sampling in those regions associated with a higher intensity. Finally, the reconstructed signal is sampled on a regular grid around the input domain, and its behaviour is represented through the variation of the corresponding iso-surfaces.

For the previous tests, we evaluate the approximation accuracy by analysing the cumulative error (c.f., Eq. (4)) of the reconstructed signal (Fig. 12). We solve the PDE on: (i) the original input points; (ii) the n samples computed with the kernel-based method; and (iii) n points uniformly distributed in the domain. We order the points in (i) on decreasing order with respect to their intensity value. Then, for each point in (i), we compute the error between its solution and the solution of the closest point in (ii) and (iii), respectively. In both 2D (a) and 3D (b) cases, the solution (ii) has a very low error on the first input points, i.e., the ones associated with a higher intensity signal. In the 2D case, the cumulative error computed on the first 4777 points (which covers from the 47% up to the 100% of the intensity of the signal) is 0.01. The solution (iii) on the same 4777 points has an error equal to 0.033. In the 3D case, the cumulative error computed on the first 1900 points (which covers from the 27% up to the 100% of the intensity of the signal) is 1.6. The solution (iii) on the

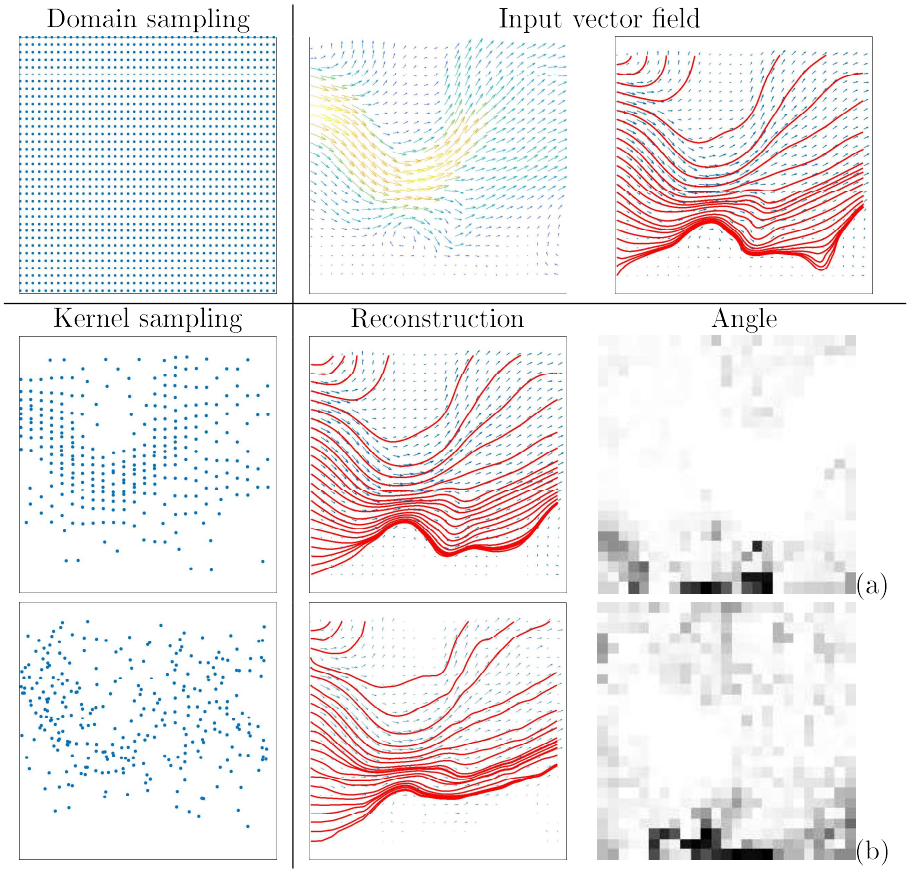


Figure 13: (First row) Input wind field and streamlines on a regular grid with $m = 625$ points. Kernel-based sampling with $n = 300$ samples, reconstructed vector field and streamlines, variation of the angle variation between the input and the reconstructed field: (a) ($\boldsymbol{\sigma}$,$\boldsymbol{\alpha}$) kernel-based sampling and (b) least-squares potential sampling. White represents a null angle and black corresponds to a maximum angle of π.

same 1900 points has an error equal to 11.1. Indeed, our sampling improves the approximation of the input signal in those regions where the intensity is higher.

5.3. Kernel-based sampling of vector fields

We apply the kernel-based sampling to a vector field $\mathbf{v}(\mathbf{x}) : \Omega \to \mathbb{R}^d$ defined on an arbitrary input domain by considering its energy or components.

As first option (*energy-based sampling*), we apply the proposed ($\boldsymbol{\sigma}$,$\boldsymbol{\alpha}$)-method (Sect. 3.1) to the energy function $f(\mathbf{x}) := \|\mathbf{v}(\mathbf{x})\|_2$ of the input vector field in

Table 11: With reference to Fig. 13, we report the approximation accuracy of the three methods introduced in Sect. 5.3, applied to a vector field defined on a regular grid, with best results in bold.

Option	1^{st} opt.	2^{nd} opt.
Method	Potential sampl.	Least-squares potential sampl.
$\overline{\mathbf{NCC}}$	**0.994**	0.912
$\overline{\mathbf{NRMSE}}$	**0.084**	0.331
$\overline{P_{0.05}}$	**99.9%**	85.7%
$\overline{P_{0.10}}$	**100%**	96.9%

Table 12: With reference to Fig. 14, we report the approximation accuracy of the three methods introduced in Sect. 5.3, applied to a vector field defined on an irregular grid, with best results in bold.

Option	1^{st} opt.	2^{nd} opt.
Method	Potential sampl.	Least-squares potential sampl.
$\overline{\mathbf{NCC}}$	0.9877	**0.9967**
$\overline{\mathbf{NRMSE}}$	0.1525	**0.0804**
$\overline{P_{0.05}}$	98.7%	**99.6%**
$\overline{P_{0.10}}$	99.7%	**99.9%**

order to compute the set of variables that are then used to recover the potential $F(\mathbf{x}) := \sum_{j=1}^{n} \alpha_j G(\mathbf{x}, \boldsymbol{\mu}_j, \sigma_j)$ of $\mathbf{v}(\mathbf{x})$. Then, the irrotational component of the input vector field is approximated as ∇F. Alternatively, we approximate each component $\mathbf{v}_i : \Omega \to \mathbb{R}$ of $\mathbf{v}$ through the least-squares scheme (Eq. (5)).

As second option (*least-squares potential sampling*), we define the energy functional as $E(\boldsymbol{\mu}, \boldsymbol{\alpha}, \boldsymbol{\sigma}) = \sum_{h=1}^{d} \int_{\Omega} |v_h(\mathbf{x}) - k_h \sum_{j=1}^{n} \alpha_{hj} G(\mathbf{x}, \boldsymbol{\mu}_j, \sigma_{hj})|^2 d\mathbf{s}$, and the function $G(\mathbf{x}, \boldsymbol{\mu}_j, \sigma_{hj})$ is defined according to Eq. (1). The derivatives are

$$\frac{\partial E(\boldsymbol{\mu}, \boldsymbol{\alpha}, \boldsymbol{\sigma})}{\partial \boldsymbol{\mu}_i} = \sum_{h=1}^{d} \frac{-2k_h}{(\sqrt{2\pi})^d \sigma_{hi}^{d+2}} \alpha_{hi} \int_{\Omega} \bar{S}_h(\mathbf{x}, \boldsymbol{\mu}, \boldsymbol{\sigma}, \boldsymbol{\alpha}) \exp(-\frac{\|\mathbf{x} - \boldsymbol{\mu}_{hi}\|_2^2}{2\sigma_{hi}^2})(\mathbf{x} - \boldsymbol{\mu}_i) d\mathbf{s};$$

$$\frac{\partial E(\boldsymbol{\mu}, \boldsymbol{\alpha}, \boldsymbol{\sigma})}{\partial \alpha_{hi}} = \frac{-2k}{(\sqrt{2\pi}\sigma_{hi})^d} \int_{\Omega} \bar{S}_h(\mathbf{x}, \boldsymbol{\mu}, \boldsymbol{\sigma}, \boldsymbol{\alpha}) \exp(-\frac{\|\mathbf{x} - \boldsymbol{\mu}_i\|_2^2}{2\sigma_{hi}^2}) d\mathbf{s};$$

$$\frac{\partial E(\boldsymbol{\mu}, \boldsymbol{\alpha}, \boldsymbol{\sigma})}{\partial \sigma_{hi}} = \frac{-2k}{(\sqrt{2\pi}\sigma_{hi})^d} \int_{\Omega} \bar{S}_h(\mathbf{x}, \boldsymbol{\mu}, \boldsymbol{\sigma}, \boldsymbol{\alpha}) \exp(-\frac{\|\mathbf{x} - \boldsymbol{\mu}_i\|_2^2}{2\sigma_{hi}^2}) \left[\frac{-d}{\sigma_{hi}} + \frac{\|\mathbf{x} - \boldsymbol{\mu}_i\|_2^2}{\sigma_{hi}^3}\right] d\mathbf{s},$$

where $\bar{S}_h^2(\mathbf{x}, \boldsymbol{\mu}, \boldsymbol{\sigma}, \boldsymbol{\alpha})$ is the h-th integrand term of the energy functional. Then, the i-th component of the vector field is approximated as $k_i \sum_{j=1}^{n} \alpha_{ij} G(\mathbf{x}, \boldsymbol{\mu}_j, \sigma_{ij})$, or through the least-squares scheme.

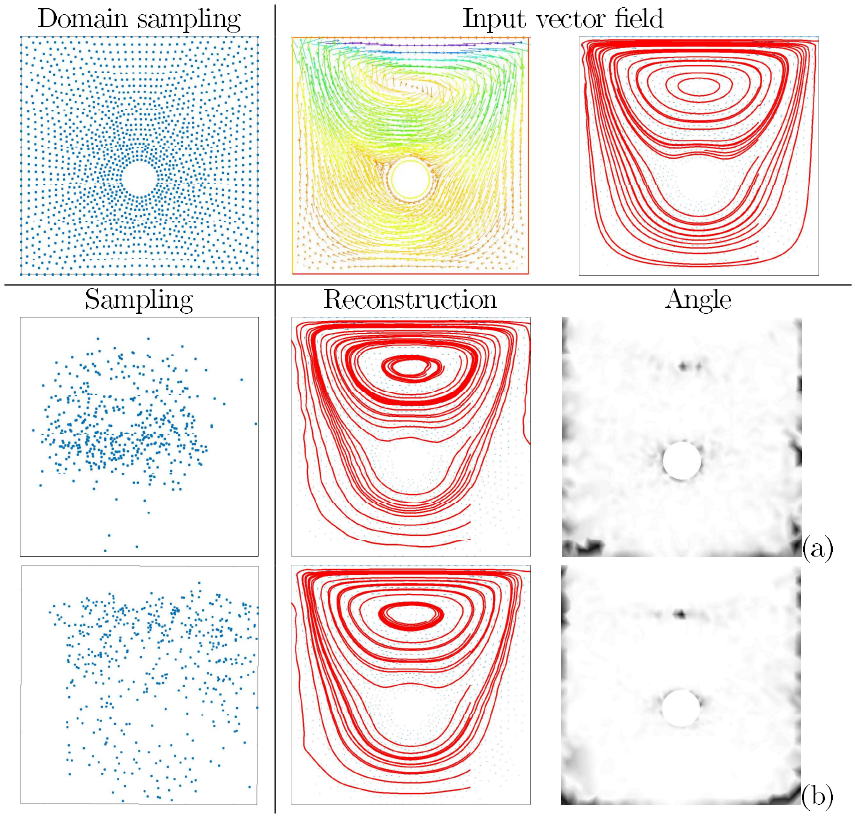


Figure 14: (First row) Irregularly-sampled water flow field ($m = 1500$ input points, $n = 500$ samples) and its streamlines. (Second-third rows) Kernel-based sampling, reconstruction, and angle variation between the input and approximated vector field with respect to (a) the energy-based sampling and the (b) least-squares potential sampling. White represents a null angle and black corresponds to a maximum angle of π.

Results. We apply the two kernel-based sampling methods to 2D vector fields: the first one (Fig. 13) is a wind field defined on a regular grid and the second one (Fig. 14) is a fluid flow field on an irregularly-sampled domain. The error is measured as the angle between the input and the reconstruction vector field; white corresponds to a null angle and black identifies the maximum angle, which is lower or equal to π in our experiments (Figs. 13, 14, 15). For all the methods, the samples are denser in those regions where the intensity of the vector field is higher, the reconstruction is very accurate and preserves the flow lines of the vector field. It is more precise where the intensity of the signal is higher;

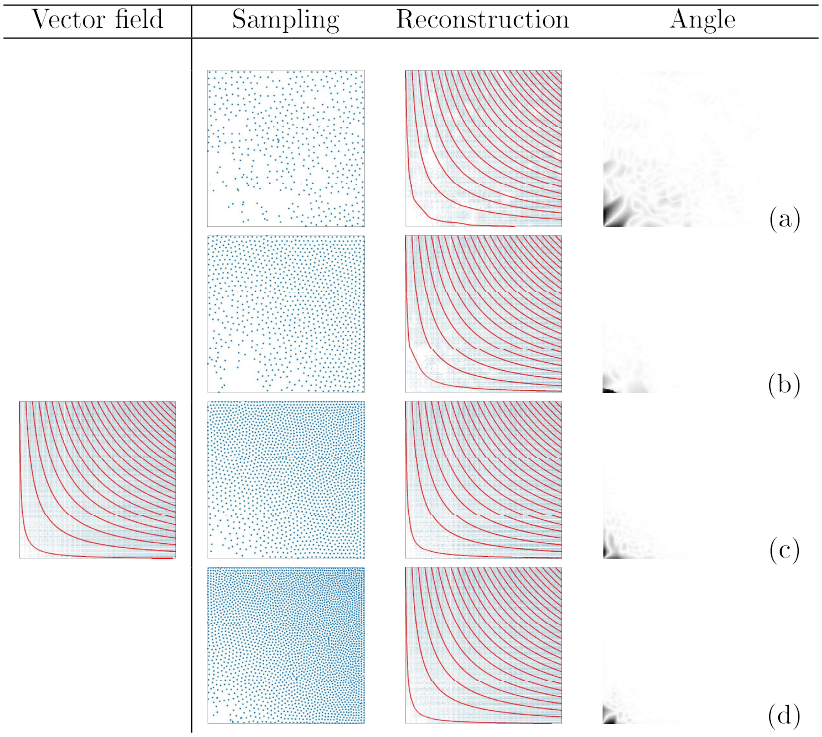


Figure 15: Sampling, reconstruction, and variation of the angle of the kernel-based sampling applied to the vector field, varying the number of samples. (a) $n = 500$; (b) $n = 1000$; (c) $n = 2000$; (d) $n = 3000$; $m = 128 \times 128$ input points. White represents a null angle and black corresponds to a maximum angle of π.

for example, in the central region of the input domain (wind case), or in the region around the circular obstacle (fluid case). The error is generally low, and it is mainly localised in the boundary regions, where we have only a partial information on the behaviour of the input vector field.

Approximation accuracy. We compare the error metrics for the three methods, by computing the average of each metrics (i.e., $\overline{\text{NCC}}$, $\overline{\text{NRMSE}}$, $\overline{P_{0.05}}$, $\overline{P_{0.10}}$) on all the components of the vector field, e.g., $\overline{\text{NCC}} = \sum_{i=1}^{d} \text{NCC}_i/d$, where NCC_i is the normalised cross correlation NCC metric in Eq. (3) of the reconstruction of the i-th component of the vector field. On the regular grid case (Table 11), the first method has the best results, with a $\overline{\text{NCC}}$ value of 0.994 and a $\overline{P_{0.05}}$ value of 99.9%, while the second approach has a $\overline{\text{NCC}}$ value of 0.912. On the

Table 13: With reference to Fig. 15, we report the error metrics for the first method when varying the number of samples, with best results in bold.

Samples	4K	3K	2K	1K	0.5K
Objective function	**2.98**	7.19	26.7	293.6	1656
$\overline{\textbf{NCC}}$	**0.999**	0.998	0.995	0.981	0.913
$\overline{\textbf{NRMSE}}$	**0.020**	0.029	0.048	0.102	0.226
$\overline{P_{0.05}}$	**100%**	99.9%	98.4%	88.2%	67.1%
$\overline{P_{0.10}}$	**100%**	100%	99.9%	97.1%	85.1%

Table 14: With reference to Fig. 16, we report the error metrics for the first method when varying the number of samples, with best results in bold.

Samples	700	500	300	150
Objective function	**1.49**	1.81	4.6	8.3
$\overline{\textbf{NCC}}$	**0.997**	0.996	0.983	0.959
$\overline{\textbf{NRMSE}}$	**0.075**	0.152	0.181	0.279
$\overline{P_{0.05}}$	**99.9%**	98.7%	98.6%	96.1%
$\overline{P_{0.10}}$	**100%**	99.7%	99.6%	99.1%

irregular grid case (Table 12), the second method has the best results, with a $\overline{\text{NCC}}$ value of 0.9967 and a $\overline{P_{0.05}}$ value of 99.6%. The first method has also very good results, with a $\overline{\text{NCC}}$ value of 0.9877.

Accuracy with respect to the number of samples. Since the first variant has been identified as the best in terms of approximation accuracy, we further analyse its accuracy with respect to the number of samples. Considering a regularly-sampled vector field and selecting a different number of samples (i.e., from 500 to 4K), the accuracy of the reconstructed vector field improves and the error is mainly localised at the bottom left corner, where the samples are less dense, and it reduces where the number of samples increases (Fig. 15). Comparing the error metrics with a different number of samples (Table 13), the ($\boldsymbol{\sigma}$,$\boldsymbol{\alpha}$)-method is very accurate, even when we use only 1K samples (i.e., the 6% of the input points). In fact, the 97.1% of the points have a reconstruction error lower than 0.10. Selecting 4K samples, the 100% of the points have a reconstruction error lower than 0.05. Considering an irregularly-sampled vector field (Fig. 16) and comparing the error metrics with a different number of samples (Table 14), the kernel-based sampling is very accurate, even when we use only 150 samples

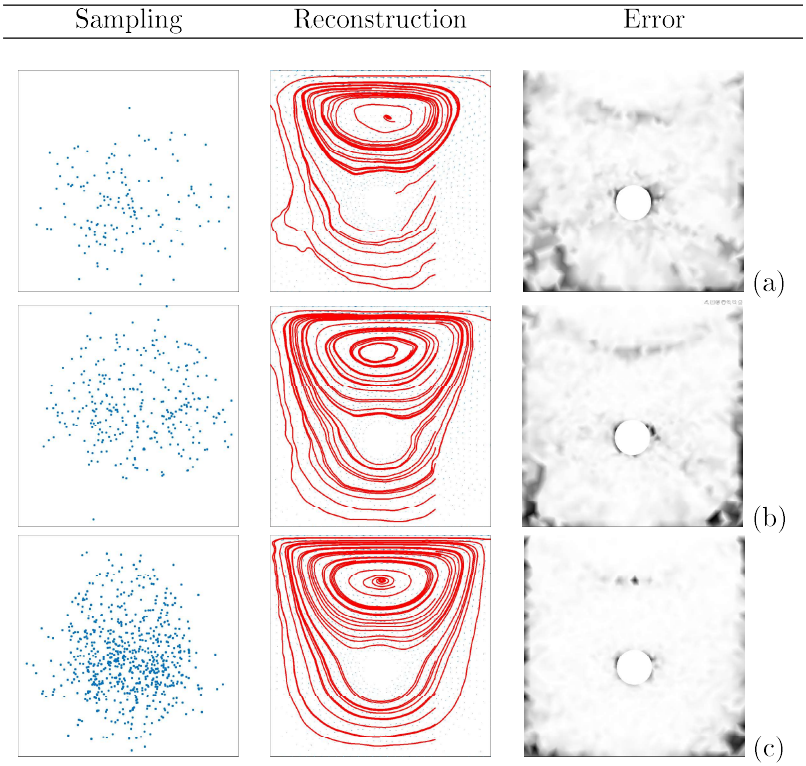


Figure 16: Sampling, reconstruction, and error of the kernel-based sampling applied to the vector field introduced in Fig. 14, varying the number of samples. (a) $n = 150$; (b) $n = 300$; (c) $n = 700$.

(i.e., the 10% of the input points). In fact, the 96.1% of the points have a reconstruction error lower than 0.05. Selecting 700 samples, the 100% of the points have a reconstruction error lower than 0.10. In Tables 15, 16, we report the execution time and the number of iterations with respect to a different number of selected samples.

5.4. Limitations

Evaluating the reconstructed signal representation (Eq. (1)) at a new set of evaluation points (e.g., at the nodes of a higher or lower resolution grid), we can upsample or downsample the input signal. For instance, the 2× upsampling of Supermario and Puccini (Fig. 17) is very smooth on uniform grey regions, and well preserves the image features; the reconstruction error is also very low,

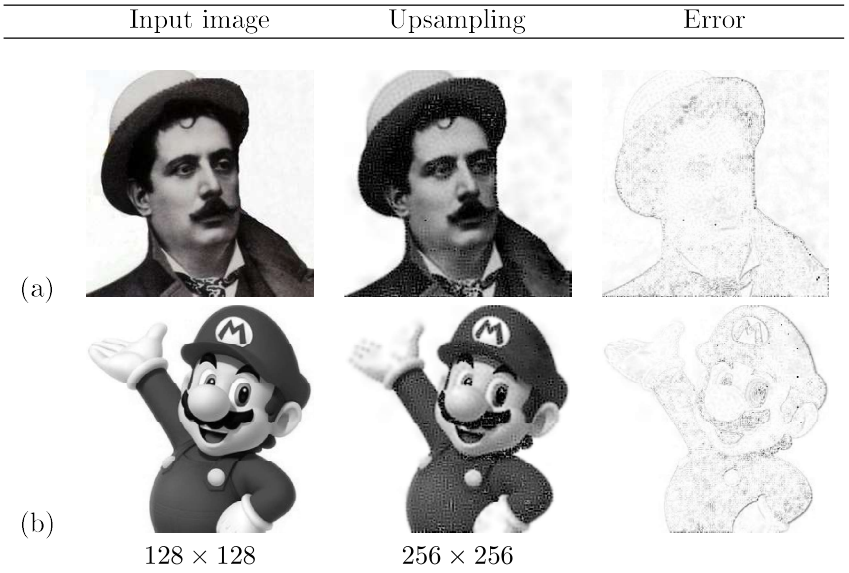


Figure 17: Upsampling and reconstruction error of (a) Puccini, and (b) Supermario, with an upsampling from 128×128 to 256×256, with 5K samples.

without noise on the background. The dot-like effect is slightly visible on the tie and on the hair of Puccini, and on the moustaches of Supermario; in fact, the error is higher in those regions. According to Table 17, the approximation accuracy is very good for both the examples; the NCC value is 0.944 and 0.961 for the Supermario and Puccini images respectively. Furthermore, both the reconstructed images approximate more than the 90% of the input points with an error lower than 0.1. The dot-like effect in Fig. 17 is induced by the noise in the background, which is then reproduced in the approximation and upsampling, and represents a possible limitation of the upsampling.

6. Conclusions and future work

We have presented different novel variants of the kernel-based sampling, by optimising the centres, supports, and coefficients of the reconstructed signal. Our method can be applied to any signal defined on 2D and 3D data (e.g., images, simulations on unstructured grids, vector-valued functions) and allows us to improve the signal approximation and reconstruction, by preserving the

Table 15: With reference to Fig. 15, we report the reduction of the execution time and the variation of the number of iterations for the kernel-based sampling with a different number of samples, with respect to the execution time $T = 1240s$ with 4K samples.

Samples	4K	3K	2K	1K	0.5K
Execution Time [s]	T	$0.74T$	$0.48T$	$0.23T$	$0.09T$
Iterations	454	573	576	531	438

Table 16: With reference to Fig. 16, we report the reduction of the execution time and the variation of the number of iterations for the kernel-based sampling with a different number of samples, with respect to the execution time $T = 58s$ with 700 samples.

Samples	700	500	300	150
Execution Time [s]	T	$0.4T$	$0.23T$	$0.22T$
Iterations	1634	842	631	1236

original behaviour in terms of features and grey intensities for 2D images and accuracy for arbitrary data (e.g., solution to PDEs, vector fields).

The additional number of variables of our method forces us to introduce a sampling initialisation, when dealing with high-resolution images. However, this approach shows some limits: the clustering effect of the samples on the overlapped area is difficult to manage and averaging the positions of the samples does not achieve an optimal sampling, in terms of feature preservation. In this last case, clustering with edge-preservation features and image stitching method [53] are necessary to better overcome the overlap of patches in the reconstructed images.

As main future work, we plan to further investigate the upsampling through a filtering of the image before the sampling and upsampling steps and the use of interpolating constraints along feature lines. Since the main limitation of our method is the higher computational cost with respect to the original kernel-based sampling, as a matter of a higher number of variables, another future activity is the reduction of the execution time of our method (e.g., through parallelisation) in order to address real-time applications (e.g., 2D video processing).

Acknowledgements. We thank the Reviewers for their comments, which helped us to improve the technical and experimental parts of the article.

Table 17: Approximation accuracy of the upsampling for the example in Fig. 17.

Image	Supermario	Puccini
NCC	0.944	0.961
NRMSE	0.158	0.1616
$P_{0.05}$	76.6%	76.7%
$P_{0.10}$	90.1%	90.4%

Appendix I - Energy functional and derivatives

We introduce the energy functional and the derivatives of the variants of the kernel-based sampling introduced in Sect. 3.1.

(σ)-*method.* The energy functional is defined as

$$E(\boldsymbol{\mu}, \sigma) = \int_\Omega |C(\mathbf{x}) - k \sum_{j=1}^{n} G(\mathbf{x}, \boldsymbol{\mu}_j, \sigma)|^2 d\mathbf{s},$$

and its derivatives are

$$\frac{\partial E(\boldsymbol{\mu},\sigma)}{\partial \boldsymbol{\mu}_i} = \frac{-2k}{(\sqrt{2\pi})^d \sigma^{d+2}} \int_\Omega [\bar{S}(\mathbf{x}, \boldsymbol{\mu}, \sigma) \exp(-\frac{\|\mathbf{x}-\boldsymbol{\mu}_i\|_2^2}{2\sigma^2})(\mathbf{x} - \boldsymbol{\mu}_i)] d\mathbf{s};$$
$$\frac{\partial E(\boldsymbol{\mu},\sigma)}{\partial \sigma} = \frac{-2k}{(\sqrt{2\pi}\sigma)^d} \int_\Omega \bar{S}(\mathbf{x}, \boldsymbol{\mu}, \sigma) \sum_{j=1}^{n} \exp(-\frac{\|\mathbf{x}-\boldsymbol{\mu}_j\|_2^2}{2\sigma^2})[\frac{-d}{\sigma} + \frac{-\|\mathbf{x}-\boldsymbol{\mu}_j\|_2^2}{\sigma^3}] d\mathbf{s}.$$

($\boldsymbol{\sigma}$)-*method.* The Gaussian kernel is defined as

$$G(\mathbf{x}, \boldsymbol{\mu}_j, \sigma_j) = \frac{1}{(\sqrt{2\pi}\sigma_j)^d} \exp(-\frac{\|\mathbf{x} - \boldsymbol{\mu}_j\|_2^2}{2\sigma_j^2}).$$

The energy functional is defined as

$$E(\boldsymbol{\mu}, \boldsymbol{\sigma}) = \int_\Omega |C(\mathbf{x}) - k \sum_{j=1}^{n} G(\mathbf{x}, \boldsymbol{\mu}_j, \sigma_j)|^2 d\mathbf{s},$$

and its derivatives are

$$\frac{\partial E(\boldsymbol{\mu},\boldsymbol{\sigma})}{\partial \boldsymbol{\mu}_i} = \frac{-2k}{(\sqrt{2\pi})^d \sigma_i^{d+2}} \int_\Omega [\bar{S}(\mathbf{x}, \boldsymbol{\mu}, \boldsymbol{\sigma}) \exp(-\frac{\|\mathbf{x} - \boldsymbol{\mu}_i\|_2^2}{2\sigma_i^2})(\mathbf{x} - \boldsymbol{\mu}_i)] d\mathbf{s};$$
$$\frac{\partial E(\boldsymbol{\mu},\boldsymbol{\sigma})}{\partial \sigma_i} = \frac{-2k}{(\sqrt{2\pi}\sigma_i)^d} \int_\Omega \bar{S}(\mathbf{x}, \boldsymbol{\mu}, \boldsymbol{\sigma}) \exp(-\frac{\|\mathbf{x} - \boldsymbol{\mu}_i\|_2^2}{2\sigma_i^2})[\frac{-d}{\sigma_i} + \frac{-\|\mathbf{x} - \boldsymbol{\mu}_i\|_2^2}{\sigma_i^3}] d\mathbf{s}.$$

($\boldsymbol{\alpha}$)-*method.* The energy functional is defined as

$$E(\boldsymbol{\mu}, \boldsymbol{\alpha}) = \int_\Omega |C(\mathbf{x}) - k \sum_{j=1}^{n} \alpha_j G(\mathbf{x}, \boldsymbol{\mu}_j)|^2 d\mathbf{s},$$

and its derivatives are

$$\frac{\partial E(\boldsymbol{\mu},\boldsymbol{\alpha})}{\partial \boldsymbol{\mu}_i} = \frac{-2k}{(\sqrt{2\pi})^d \sigma^{d+2}} \alpha_i \int_\Omega \bar{S}(\mathbf{x},\boldsymbol{\mu},\boldsymbol{\alpha}) \exp(-\frac{\|\mathbf{x}-\boldsymbol{\mu}_i\|_2^2}{2\sigma^2})(\mathbf{x}-\boldsymbol{\mu}_i) d\mathbf{s};$$

$$\frac{\partial E(\boldsymbol{\mu},\boldsymbol{\alpha})}{\partial \alpha_i} = \frac{-2k}{(\sqrt{2\pi}\sigma)^d} \int_\Omega \bar{S}(\mathbf{x},\boldsymbol{\mu},\boldsymbol{\alpha}) \exp(-\frac{\|\mathbf{x}-\boldsymbol{\mu}_i\|_2^2}{2\sigma^2}) d\mathbf{s}.$$

References

[1] Analoui, M., Allebach, J., 1992. New results on reconstruction of continuous-tone from halftone, in: Proc. of the IEEE International Conf. on Acoustics, Speech, and Signal Processing, pp. 313–316.

[2] Balzer, M., Schlömer, T., Deussen, O., 2009. Capacity-constrained point distributions: a variant of lloyd's method. ACM Trans. on Graphics 28.

[3] Biernacki, C., Celeux, G., Govaert, G., 2000. Assessing a mixture model for clustering with the integrated completed likelihood. IEEE Trans. on Pattern Analysis and Machine Intelligence 22, 719–725.

[4] Bossen, F., 1996. Anisotropic Mesh Generation with Particles. Technical Report. Carnegie-Mellon Univ. Pittsburgh PA Dept. of Computer Science.

[5] Bowers, J., Wang, R., Wei, L.Y., Maletz, D., 2010. Parallel Poisson disk sampling with spectrum analysis on surfaces, in: ACM Trans. on Graphics, p. 166.

[6] Candocia, F.M., Principe, J.C., 1999. Super-resolution of images based on local correlations. IEEE Trans. on Neural Networks 10, 372–380.

[7] Chen, J., Ge, X., Wei, L.Y., Wang, B., Wang, Y., Wang, H., Fei, Y., Qian, K.L., Yong, J.H., Wang, W., 2013. Bilateral blue noise sampling. ACM Trans. on Graphics 32, 216.

[8] Chen, Z., Yuan, Z., Choi, Y.K., Liu, L., Wang, W., 2012. Variational blue noise sampling. IEEE Trans. on Visualization and Computer Graphics 18, 1784–1796.

[9] Chen, Z., Zhang, T., Cao, J., Zhang, Y.J., Wang, C., 2018. Point cloud resampling using centroidal Voronoi tessellation methods. Computer-Aided Design 102, 12–21.

[10] Chu, H.K., Chang, C.S., Lee, R.R., Mitra, N.J., 2013. Halftone QR codes. ACM Trans. on Graphics 32, 217.

[11] Cook, R.L., 1986. Stochastic sampling in Computer Graphics. ACM Trans. on Graphics 5, 51–72.

[12] De Goes, F., Breeden, K., Ostromoukhov, V., Desbrun, M., 2012. Blue noise through optimal transport. ACM Trans. on Graphics 31, 1–11.

[13] Debnath, J., Parrinello, M., 2020. Gaussian mixture-based enhanced sampling for statics and dynamics. The Journal of Physical Chemistry Letters 11, 5076–5080.

[14] Deussen, O., Hiller, S., Van Overveld, C., Strothotte, T., 2000. Floating points: A method for computing stipple drawings, in: Computer Graphics Forum, pp. 41–50.

[15] Dippé, M.A., Wold, E.H., 1985. Antialiasing through stochastic sampling. ACM Siggraph Computer Graphics 19, 69–78.

[16] Donohue, C., Ostromoukhov, V., 2007. Fast generation of importance-sampled point sets with associated delaunay triangulation, in: Proc. of GRAPHICON, Citeseer. pp. 125–130.

[17] Dunbar, D., Humphreys, G., 2006. A spatial data structure for fast Poisson-disk sample generationpoisson-disk sample generation, in: ACM Trans. on Graphics, pp. 503–508.

[18] Fabritius, B., Tabor, G., 2016. Improving the quality of finite volume meshes through genetic optimisation. Engineering with Computers 32, 425–440.

[19] Farrugia, J.P., Péroche, B., 2004. A progressive rendering algorithm using an adaptive perceptually based image metric, in: Computer Graphics Forum, pp. 605–614.

[20] Fattal, R., 2011. Blue-noise point sampling using kernel density model, in: ACM Trans. on Graphics, p. 48.

[21] Fruchter, A., Hook, R., 2002. Drizzle: A method for the linear reconstruction of undersampled images. Publications of the Astronomical Society of the Pacific 114, 144.

[22] Fu, L., Han, L., Hu, X.Y., Adams, N.A., 2019. An isotropic unstructured mesh generation method based on a fluid relaxation analogy. Computer Methods in Applied Mechanics and Engineering 350, 396–431.

[23] Greenspan, H., Ruf, A., Goldberger, J., 2006. Constrained Gaussian mixture model framework for automatic segmentation of MR brain images. IEEE Trans. on Medical Imaging 25, 1233–1245.

[24] Hanebeck, U.D., 2014. Kernel-based deterministic blue-noise sampling of arbitrary probability density functions, in: Conf. on Information Sciences and Systems, IEEE. pp. 1–6.

[25] Heck, D., Schlömer, T., Deussen, O., 2013. Blue noise sampling with controlled aliasing. ACM Trans. on Graphics 32, 1–12.

[26] Keller, A., Heidrich, W., 2001. Interleaved sampling, in: Rendering Techniques 2001. Springer, pp. 269–276.

[27] Kong, A., McCullagh, P., Meng, X.L., Nicolae, D., Tan, Z., 2003. A theory of statistical models for monte carlo integration. Journal of the Royal Statistical Society: Series B 65, 585–604.

[28] Kopf, J., Cohen-Or, D., Deussen, O., Lischinski, D., 2006. Recursive wang tiles for real-time blue noise, in: ACM Siggraph, pp. 509–518.

[29] Kurtz, N., Song, J., 2013. Cross-entropy-based adaptive importance sampling using Gaussian mixture. Structural Safety 42, 35–44.

[30] Lau, D.L., Arce, G.R., 2001. Modern digital halftoning. CRC Press.

[31] Li, X., 2006. Edge-directed error diffusion halftoning. IEEE Signal Processing Letters 13, 688–690.

[32] Lloyd, S., 1982. Least-squares quantization in pcm. IEEE Trans. on Information Theory 28, 129–137.

[33] Maugis, C., Celeux, G., Martin-Magniette, M.L., 2009. Variable selection for clustering with Gaussian mixture models. Biometrics 65, 701–709.

[34] Mérigot, Q., 2011. A multiscale approach to optimal transport, in: Computer Graphics Forum, Wiley Online Library. pp. 1583–1592.

[35] Mitchell, D.P., 1987. Generating antialiased images at low sampling densities. ACM SIGGRAPH Computer Graphics 21, 65–72.

[36] Ostromoukhov, V., Hersch, R.D., 1995. Artistic screening. Technical Report. ACM.

[37] Öztireli, A.C., Alexa, M., Gross, M., 2010. Spectral sampling of manifolds. ACM Trans. on Graphics 29, 168.

[38] Pang, W.M., Qu, Y., Wong, T.T., Cohen-Or, D., Heng, P.A., 2008. Structure-aware halftoning, in: ACM Trans. on Graphics, p. 89.

[39] Pattanaik, S.N., Ferwerda, J.A., Fairchild, M.D., Greenberg, D.P., 1998. A multiscale model of adaptation and spatial vision for realistic image display, in: Proc. of Conf. on Computer Graphics and Interactive Techniques, Citeseer. pp. 287–298.

[40] Paulin, L., Bonneel, N., Coeurjolly, D., Iehl, J.C., Webanck, A., Desbrun, M., Ostromoukhov, V., 2020. Sliced optimal transport sampling. ACM Trans. on Graphics 39.

[41] Pauly, M., Keiser, R., Kobbelt, L.P., Gross, M., 2003. Shape modeling with point-sampled geometry, in: ACM Trans. on Graphics, pp. 641–650.

[42] Persson, P.O., Strang, G., 2004. A simple mesh generator in MATLAB. SIAM Review 46, 329–345.

[43] Pharr, M., Jakob, W., Humphreys, G., 2016. Physically based rendering: From theory to implementation. Morgan Kaufmann.

[44] Qin, H., Chen, Y., He, J., Chen, B., 2017. Wasserstein blue noise sampling. ACM Trans. on Graphics 36, 1–13.

[45] Raftery, A.E., Bao, L., 2010. Estimating and projecting trends in HIV/AIDS generalized epidemics using incremental mixture importance sampling. Biometrics 66, 1162–1173.

[46] Ravikumar, N., Gooya, A., Çimen, S., Frangi, A.F., Taylor, Z.A., 2018. Group-wise similarity registration of point sets using student's t-mixture model for statistical shape models. Medical Image Analysis 44, 156–176.

[47] Sainz, M., Pajarola, R., 2004. Point-based rendering techniques. Computers & Graphics 28, 869–879.

[48] Schmaltz, C., Gwosdek, P., Bruhn, A., Weickert, J., 2010. Electrostatic halftoning, in: Computer Graphics Forum, pp. 2313–2327.

[49] Secord, A., 2002. Weighted Voronoi stippling, in: Proc. of the Symposium on non-photorealistic animation and rendering, pp. 37–43.

[50] Sheather, S.J., 2004. Density estimation. Statistical science , 588–597.

[51] Shimada, K., Gossard, D.C., 1995. Bubble mesh: automated triangular meshing of non-manifold geometry by sphere packing, in: Proc. of the ACM Symposium on Solid Modeling and Applications, pp. 409–419.

[52] Su, T., Dy, J.G., 2007. In search of deterministic methods for initializing k-means and Gaussian mixture clustering. Intelligent Data Analysis 11, 319–338.

[53] Szeliski, R., 2006. Image alignment and stitching: A tutorial. Foundations and Trends® in Computer Graphics and Vision 2, 1–104.

[54] Teodoro, A.M., Almeida, M.S., Figueiredo, M.A., 2015. Single-frame image denoising and inpainting using gaussian mixtures., in: ICPRAM (2), pp. 283–288.

[55] Terzopoulos, D., Vasilescu, M., 1991. Sampling and reconstruction with adaptive meshes, in: Proc. of Conf. on Computer Vision and Pattern Recognition, pp. 70–75.

[56] White, K.B., Cline, D., Egbert, P.K., 2007. Poisson disk point sets by hierarchical dart throwing, in: IEEE Symposium on Interactive Ray Tracing, pp. 129–132.

[57] Yang, M.S., Lai, C.Y., Lin, C.Y., 2012. A robust em clustering algorithm for Gaussian mixture models. Pattern Recognition 45, 3950–3961.

[58] Yellott, J.I., 1983. Spectral consequences of photoreceptor sampling in the rhesus retina. Science 221, 382–385.

[59] Yoon, J.H., Kim, D.Y., Yoon, K.J., 2013. Gaussian mixture importance sampling function for unscented SMC-PHD filter. Signal Processing 93, 2664–2670.

[60] Yu, G., Sapiro, G., Mallat, S., 2011. Solving inverse problems with piecewise linear estimators: From Gaussian mixture models to structured sparsity. IEEE Trans. on Image Processing 21, 2481–2499.

[61] Yu, G., Sapiro, G., Mallat, S., 2012. Solving inverse problems with piecewise linear estimators: From Gaussian mixture models to structured sparsity. IEEE Trans. on Image Processing 21, 2481–2499.

[62] Zhong, S., Zhong, Z., Hua, J., 2019. Surface reconstruction by parallel and unified particle-based resampling from point clouds. Computer-Aided Geometric Design 71, 43–62.

[63] Zhong, Z., Guo, X., Wang, W., Lévy, B., Sun, F., Liu, Y., Mao, W., et al., 2013. Particle-based anisotropic surface meshing. ACM Trans. on Graphics 32, 99–1.

[64] Zhong, Z., Hua, J., 2016. Kernel-based adaptive sampling for image reconstruction and meshing. Computer-Aided Geometric Design 43, 68–81.

[65] Zhu, C., Byrd, R.H., Lu, P., Nocedal, J., 1997. Algorithm 778: L-bfgs-b: Fortran subroutines for large-scale bound-constrained optimization. ACM Trans. on Mathematical Software 23, 550–560.