# SPLINEDIST: AUTOMATED CELL SEGMENTATION WITH SPLINE CURVES

*Soham Mandal, Virginie Uhlmann*

European Bioinformatics Institute, European Molecular Biology Laboratory, Cambridge, UK

## ABSTRACT

We present SplineDist, an instance segmentation convolutional neural network for bioimages extending the popular StarDist method. While StarDist describes objects as star-convex polygons, SplineDist uses a more flexible and general representation by modelling objects as planar parametric spline curves. Based on a new loss formulation that exploits the properties of spline constructions, we can incorporate our new object model in StarDist's architecture with minimal changes. We demonstrate in synthetic and real images that SplineDist produces segmentation outlines of equal quality than StarDist with smaller network size and accurately captures non-star-convex objects that cannot be segmented with StarDist.

***Index Terms*—** Bioimage analysis, segmentation, machine learning, convolutional neural networks, spline interpolation.

## 1. INTRODUCTION

A vast amount of biological questions investigated using microscopy data rely on an initial partitioning of the image into individual objects, a task referred to as instance segmentation [1]. Dividing an image into object regions multiplies the amount of data point that can be extracted from it. Protein expression [2] and morphology [3], among many others, can then be quantified at the single-cell level, thereby moving away from population averaging. Due to the sheer amount of imaging data acquired in modern biology experiments and because segmentation is so essential, the development of automated computational segmentation methods has been at the heart of bioimage analysis since its early days [4]. In the past decade, deep learning approaches have revolutionized this field of research [5]. Convolutional neural networks (CNN) in general, and the U-net model in particular [6], have demonstrated an unprecedented capability to consistently produce excellent segmentation results in a wide range of data, massively reducing the need for manual intervention.

Recently, StarDist [7] established itself as a popular detection and instance segmentation tool in the bioimaging community due to its simplicity and efficiency. StarDist generates object outlines as star-convex polygons: it relies on a U-net architecture to predict the locations of a fixed number of points on an object's contour, and constructs the final outline by linearly interpolating between them. This object representation is reminiscent of the one used in the parametric B-spline active contour (AC) algorithm [8] and in its many variants [9, 10, 11]. In spline AC methods, objects are segmented with a spline interpolated curve that deforms in the image following the minimization of a handcrafted energy functional. Beyond segmentation itself, the main advantage of these approaches is to provide a continuously-defined description of the object contour as a parametric curve that can readily be used to, *e.g.*, perform statistical shape analysis [12]. The flexibility of spline curves makes them suitable for a variety of segmentation problems, but their optimization requires domain-expertise, restricting their use in practice. In preliminary work, we proposed a first step towards circumventing this limitation by directly predicting spline interpolated outlines from an input image using a CNN regressor [13]. Our prototype was however limited to binary images featuring a single object.

Here, we introduce SplineDist, an instance segmentation algorithm that combines the capabilities of StarDist with those of spline approximation. Thus, SplineDist benefits both from the detection performance of StarDist and the segmentation quality of spline models. We show that SplineDist equals StarDist's excellent segmentation results with a smaller number of parameters. Besides, SplineDist can segment objects regardless of their star-convexity, thus lifting StarDist's main restriction. As a result, SplineDist expands the range of applicability of StarDist and offers a robust alternative to spline-based AC methods.

The paper is organized as follows. In Section 2, we recall the technical details of StarDist and describe the construction of SplineDist. We then experimentally compare the two methods and illustrate SplineDist's capabilities in Section 3. Finally, we conclude the paper in Section 4.

## 2. METHODS

### 2.1. StarDist

Given an input image featuring one or many biological objects, StarDist [7] predicts a star-convex polygon of $R$ vertices for every pixel. More precisely, at every pixel location $(i, j)$ in the image, StarDist predicts the set of radial distances

$D_{ij} = \{d_{ij}^k\}_{k=0,\dots,R-1}$ to the boundary of the object enclosing $(i,j)$ at $R$ equidistant angles $\frac{2\pi}{R}$ (Figure 1b). Because the distances $d_{ij}^k$ are not defined for background elements, StarDist simultaneously predicts an object probability $p_{ij}$ for each pixel, allowing to ignore proposals from pixels with low object probability. For each pixel in the image, StarDist's loss is formulated as

$$\mathcal{L}_{\text{StarDist}}(p_{ij}, \hat{p}_{ij}, D_{ij}, \hat{D}_{ij}) = \mathcal{L}_{\text{BCE}}(p_{ij}, \hat{p}_{ij}) + \lambda_1 \mathcal{L}_{\text{radii}}(p_{ij}, D_{ij}, \hat{D}_{ij}), \quad (1)$$

where $p_{ij}$ and $\hat{p}_{ij}$ are the ground truth and predicted pixel-wise object probabilities, and $D_{ij}$ and $\hat{D}_{ij}$ are the ground truth and predicted sets of radial distances to the object contour. The first term, $\mathcal{L}_{\text{BCE}}$, is the standard binary cross entropy loss [14] and $\lambda_1$ is a regularization factor. The second term, $\mathcal{L}_{\text{radii}}$, is a mean absolute error loss weighted by ground truth object probabilities and expressed as

$$\mathcal{L}_{\text{radii}}(p_{ij}, D_{ij}, \hat{D}_{ij}) = p_{ij} \cdot \mathbb{1}_{p_{ij}>0} \cdot \frac{1}{R} \sum_{k=0}^{R-1} |d_{ij}^k - \hat{d}_{ij}^k| + \lambda_2 \cdot \mathbb{1}_{p_{ij}=0} \cdot \frac{1}{R} \sum_{k=0}^{R-1} |\hat{d}_{ij}^k|, \quad (2)$$

with $\lambda_2$ a regularization factor. This loss is finally averaged over all the pixels present in the image. An important consequence of (2) is that, during training, a set of individual ground truth radial distances $D_{ij}$ must be generated for each individual pixel $(i,j)$ from a reference instance labelling of the entire image.

Once star-convex polygons have been predicted for all the pixels, overlapping candidates are filtered by a non-maximum suppression (NMS) step taking into account their associated object probabilities. The final set of polygons corresponds to individual object instances in the image.

### 2.2. SplineDist

Instead of representing objects as polygons, we propose to adopt the more flexible model used in spline AC algorithms, which uses spline interpolation. Essentially, SplineDist models an object outline as a two-dimensional parametric planar spline curve $\mathbf{s}$ defined by

$$\mathbf{s}(t) = \begin{bmatrix} s_x(t) \\ s_y(t) \end{bmatrix} = \sum_{k=0}^{M-1} \mathbf{c}[k] \varphi_M(t-k), \quad (3)$$

where $\mathbf{c}[k] = (c_x^k, c_y^k)$ are two-dimensional control points, which are the parameters of the spline model. While the number $M$ of control points is fixed, they can be located anywhere in the image. The function $\varphi_M(t) = \sum_{m\in\mathbb{Z}} \varphi(t - Mm)$ is the $M$-periodized version of a predefined spline basis $\varphi$, allowing to get a closed curve [9]. This object model is general and unifying: we can indeed consider any spline basis for $\varphi$, typically polynomial B-splines of any order. In the following, we denote as $\text{SplineDist}_{\beta^n}$ the version of SplineDist relying on $\varphi = \beta^n$, the polynomial B-spline basis of degree $n$ [15].

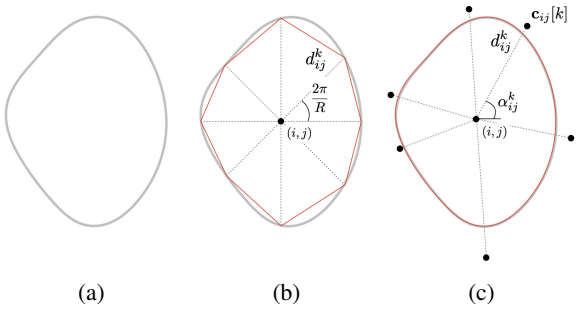



**Fig. 1**: StarDist and SplineDist object models. (a) A reference object outline. (b) For each point $(i,j)$ enclosed in the object, StarDist models the outline as a star-convex polygon of $R$ points located at fixed, uniformly distributed angles and radial distances $\{d_{ij}^k\}_{k=0,\dots,R-1}$. (c) In contrast, SplineDist models the outline as a planar spline curve (3) generated by $M$ control points $\mathbf{c}_{ij}[k] = (d_{ij}^k \cos(\alpha_{ij}^k), d_{ij}^k \sin(\alpha_{ij}^k))$. Here, $R = 8$, $M = 6$, and $\varphi = \beta^3$ the cubic B-spline basis.

Using this representation, SplineDist predicts an object probability for each image pixel $(i,j)$ along with a set of $M$ two-dimensional control points $\mathbf{c}_{ij}[k]$ that fully define the spline curve (3). Similar to the StarDist radii, the $\mathbf{c}_{ij}[k]$ are expressed in relative coordinates with respect to $(i,j)$, in polar form. More specifically, SplineDist predicts $M$ angles $\alpha_{ij}^k$ and $M$ radial distances $d_{ij}^k$ such that $\mathbf{c}_{ij}[k] = (d_{ij}^k \cos(\alpha_{ij}^k), d_{ij}^k \sin(\alpha_{ij}^k))$ (Figure 1c). This construction is more flexible than star-convex polygons: in contrast to the $R$ scalar radii at fixed angle, the $M$ two-dimensional control point vectors can point in any direction (angle and distance) from $(i,j)$. The continuously-defined curve $\mathbf{s}_{ij}$ generated by the control point sequence $\{\mathbf{c}_{ij}[k]\}_{k=0,\dots,M-1}$ can be sampled at any rate. We denote as $S_{ij} = \{\mathbf{s}_{ij}^n\}_{n=0,\dots,N-1}$ the set of $N$ discrete points obtained by uniformly sampling $\mathbf{s}_{ij}(t)$ as

$$\mathbf{s}_{ij}^n = \mathbf{s}_{ij}(t)|_{t=\frac{nM}{N}}. \quad (4)$$

It is essential that we can construct a sequence $S$ of any desired number of points $N$ from $\{\mathbf{c}_{ij}[k]\}_{k=0,\dots,M-1}$ for two reasons. First, reference annotations come in the form of instance pixel masks, from which discrete contours can efficiently be extracted during training using classical image processing methods such as Satoshi and Abe's algorithm [16]. Second, several control point sets may draw the same outline, so the very concept of ground truth control points does not make sense. We thus design SplineDist's loss to evaluate the similarity between $S_{ij}$, the discrete object outline that $\{\mathbf{c}_{ij}[k]\}_{k=0,\dots,M-1}$ generates, and a pixel-based ground truth

object outline $P_{ij} = \{\mathbf{p}_{ij}^n\}_{n=0,\dots,N-1}$ extracted from a reference instance mask. As a consequence, the value of $N$ in (4) is dictated by the number of points in the ground truth $P_{ij}$. In other words, SplineNet predicts the parameters of the continuous spline curve (3) that, when uniformly sampled, matches most closely the ground truth pixel outline of the object enclosing $(i, j)$. SplineDist's loss is formally expressed as

$$\mathcal{L}_{\text{SplineDist}}(p_{ij}, \hat{p}_{ij}, P_{ij}, S_{ij}) = \mathcal{L}_{\text{BCE}}(p_{ij}, \hat{p}_{ij}) + \lambda_1 \cdot \mathcal{L}_{\text{contour}}(p_{ij}, P_{ij}, S_{ij}) \quad (5)$$

with

$$\mathcal{L}_{\text{contour}}(p_{ij}, P_{ij}, S_{ij}) = p_{ij} \cdot \mathbb{1}_{p_{ij}>0} \cdot \frac{1}{N} \sum_{n=0}^{N-1} |\mathbf{p}_{ij}^n - \mathbf{s}_{ij}^n| + \lambda_2 \cdot \mathbb{1}_{p_{ij}=0} \cdot \frac{1}{N} \sum_{n=0}^{N-1} |\mathbf{s}_{ij}^n|, \quad (6)$$

in which $p_{ij}$ and $\hat{p}_{ij}$ are the ground truth and predicted pixel-wise object probabilities as in (1).

Because (5) involves the outline generated by the object model parameters and not the parameters themselves, a major difference with respect to StarDist is that individual pixels do not require to have their own ground truth. All pixels $(i, j)$ enclosed in the same object $\mathcal{O}$ can indeed share a single ground truth outline $P_{ij} = P_{\mathcal{O}}$ for all $(i, j) \in \mathcal{O}$ expressed in absolute (image) coordinates. The individual outline $S_{ij}$ that each pixel predicts can also straightforwardly be expressed in absolute coordinates by shifting the spline curve $\mathbf{s}_{ij}$ around $(i, j)$. This trick relieves the need to store relative coordinates describing the ground truth for each pixel. We can then afford considering the entire ground truth contour in the loss (as opposed to only $R$ points) while remaining computationally tractable. The rest of SplineDist's architecture is similar to StarDist, the remaining difference being the size of the final output layer (Figure 2).

## 3. EXPERIMENTAL RESULTS

### 3.1. Implementation Details

We use the high-quality StarDist codebase (`github.com/mpicbg-csbd/stardist`) and mainly modify modules associated with ground truth generation and loss computations. For technical reason, our implementation sets $N$ as the number of points in the largest ground truth contour in the training set, and carries out subpixel interpolation to expand all shorter ground truth contours to this size. Hence, all pixels can be processed in a single tensor in the loss without unpacking, which critically speeds up training time. Like StarDist, SplineDist is implemented in TensorFlow [17] and is available at `gitlab.ebi.ac.uk/smandal/splinedist`.

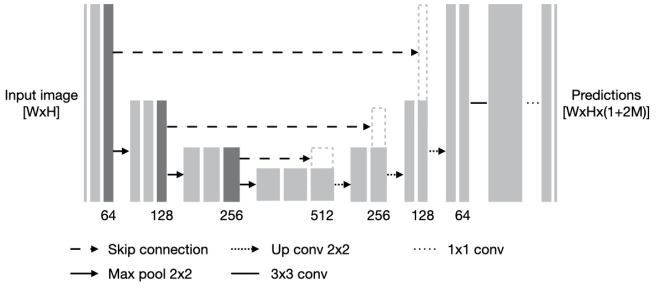



**Fig. 2**: SplineDist architecture. For each image pixel $(i, j)$, SplineDist densely predicts an object probability $p_{ij}$ and a sequence of $M$ two-dimensional control points $\{\mathbf{c}_{ij}[k]\}_{k=0,\dots,M-1}$. The overall architecture mimics StarDist up to the dimensions of the output layer, sized $W \times H \times (1 + R)$ in StarDist and $W \times H \times (1 + 2M)$ in SplineDist, with $W$ and $H$ the input image width and height, respectively.

### 3.2. Experiments

#### *3.2.1. Benchmarking*

We compare the segmentation performance of SplineDist against StarDist on the image set BBBC038v1 (also known as Kaggle 2018 Data Science Bowl dataset), available from the Broad Bioimage Benchmark Collection [18]. It is composed of a diverse collection of images of cell nuclei which faithfully reflects the variability of object appearance and image types in bioimaging. BBBC038v1 is designed to challenge the generalization capabilities of a method across these variations and is therefore widely used for benchmarking. We employ the curated subset of BBBC038v1 provided in StarDist's github repository and already used in [7] to benchmark StarDist against state-of-the-art alternatives. This dataset is composed of 447 training images and 50 testing images.

Acknowledging the good approximation properties of cubic B-splines [19], we consider SplineDist$_{\beta^3}$ in our experiments. For a fair comparison, we use StarDist's default network hyperparameters, data augmentation strategy, and training settings both in StarDist and SplineDist$_{\beta^3}$ (decaying learning rate of 0.0003, 400 epochs, batch size of 4). We also use StarDist's default NMS and object probability thresholds for both methods. We report the classical Intersection over Union (IoU) metric, also referred to as Jaccard index [20], to quantify instance segmentation quality.

In Figure 3, we quantitatively compare the performance of StarDist and SplineDist$_{\beta^3}$ with increasing equal number of points ($R = M$). As expected from spline approximation theory [21], both methods eventually converge to the same results as the number of points grows larger since StarDist can be seen as SplineDist$_{\beta^1}$ with control points at fixed angles. The performance plateau, reached by StarDist from $R = 16$ onward, is already attained by SplineDist$_{\beta^3}$ at $M = 6$. The

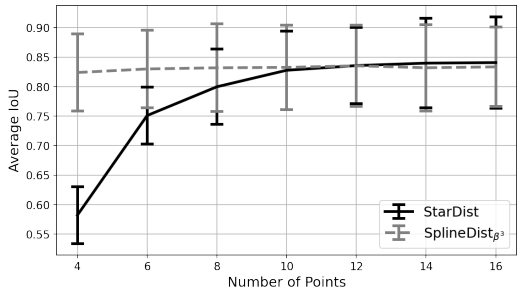
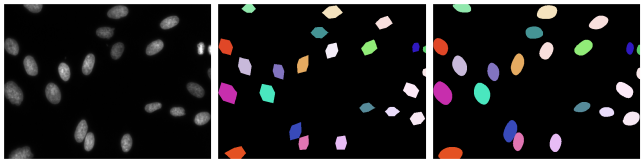



**Fig. 3**: Evolution of the average IoU on the BBBC038v1 dataset for StarDist and SplineDist$_{\beta^3}$, with object models of increasing equal number of points ($R = M$). Error bars correspond to the IoU standard deviation on the whole dataset.



(a) (b) $0.79 \pm 0.05$ (c) $0.91 \pm 0.03$

**Fig. 4**: Visual comparison of instance segmentation results. (a) Original BBBC038v1 image, results obtained with (b) StarDist and (c) SplineDist$_{\beta^3}$ for $R = M = 6$. We indicate under each image its average IoU and standard deviation.

same segmentation quality can thus be obtained with $25\%$ less output parameters (since SplineDist predict $2M$ values for objects composed of $M$ points). Increasing $M$ further does not significantly improve performance. Because objects in BBBC038v1 tend to be small, large values of $M$ result in spline curves with too many degrees of freedom, even causing a slight performance degradation for $M > 12$. In Figure 4, we show visual example of results for $R = M = 6$. The smoothness granted by cubic B-spline approximation, coupled with the ability to predict the angular and the radial components of the model's parameters, boosts SplineDist$_{\beta^3}$ performance for small $M$ values.

### *3.2.2. Segmentation of non-star-convex objects*

Because it can predict control points anywhere around the object and doesn't rely on predefined angles, SplineDist can generate non-star-convex outlines. There is indeed no restriction imposing that all control points should have different angles. To explore this difference, we compare the performance of StarDist and SplineDist on a set of synthetic images containing mostly star-convex and some non-star convex cell-like objects. In that way, StarDist can still converge and we do not explicitly enforce SplineDist to learn how to represent non-star-convex objects. Our synthetic dataset is composed of 500 grayscale (8-bit) $512 \times 512$ images featuring from 1 to

(a) (b) $0.67 \pm 0.25$ (c) $0.80 \pm 0.05$ (d) $0.78 \pm 0.06$

**Fig. 5**: Segmentation of non-star-convex objects. (a) Original synthetic image, results obtained with (b) vanilla StarDist ($R = 32$), (c) SplineDist$_{\beta^1}$ ($M = 32$), and (d) SplineDist$_{\beta^3}$ ($M = 8$). We indicate under each image its average IoU and standard deviation.

20 randomly-shaped, non-overlapping deformed binary blobs that are subsequently degraded by non-uniform illumination, additive Poisson-Gaussian noise, and Gaussian blurring. We randomly split the dataset in 450 images for training and 50 for testing.

We compare the vanilla StarDist ($R = 32$) against SplineDist$_{\beta^1}$ with $M = 32$ (also a polygon of 32 vertices, but without the star-convex restriction) and SplineDist$_{\beta^3}$ with $M = 8$. The performance of SplineDist$_{\beta^1}$ reflects what could be expected from StarDist if radii angles were not fixed, while those obtained with SplineDist$_{\beta^3}$ illustrate the advantage brought by a smoother spline basis. While the average IoU of the three methods do not significantly differ since most objects in the dataset have pure or nearly star-convex shapes, a more in-depth assessment of the results (Figure 5) reveals a clear difference in performance on non-star-convex objects. StarDist either misses parts of non-star-convex objects or breaks them into several instances. We also note that SplineDist$_{\beta^3}$ achieves quantitatively equivalent and less noisy results than SplineDist$_{\beta^1}$ with $4\times$ fewer points.

## 4. CONCLUSIONS

We have introduced SplineDist, a modification of StarDist in which objects are represented as planar spline curves. SplineDist produces segmentation outlines of equally good quality as StarDist with an object model involving fewer parameters. Additionally, SplineDist generalises StarDist by lifting the star-convex polygon limitation. SplineDist therefore combines the detection performance of StarDist with the segmentation quality of spline curve models as traditionally used in parametric AC, while alleviating the weaknesses of either of these methods alone.

**Acknowledgements** The authors would like to thank Martin Weigert for inspiring discussions and helpful comments on this work, and Julien Fageot for valuable comments on the manuscript. This work is supported by EMBL core funding. The authors have no relevant financial or non-financial conflict of interest to disclose.

**Compliance with Ethical Standards** This is a computational study for which no ethical approval was required.

## 5. REFERENCES

[1] R. Szeliski, *Computer vision: algorithms and applications*, Springer-Verlag London, London, UK, 2011.

[2] D. Aymoz, V. Wosika, E. Durandau, and S. Pelet, "Real-time quantification of protein expression at the single-cell level via dynamic protein synthesis translocation reporters," *Nature Communications*, vol. 7, no. 1, pp. 1–12, 2016.

[3] P.-H. Wu et al., "Single-cell morphology encodes metastatic potential," *Science Advances*, vol. 6, no. 4, pp. eaaw6938, 2020.

[4] E. Meijering, "Cell segmentation: 50 years down the road," *IEEE Signal Processing Magazine*, vol. 29, no. 5, pp. 140–145, 2012.

[5] E. Meijering, "A bird's-eye view of deep learning in bioimage analysis," *Computational and Structural Biotechnology Journal*, vol. 18, pp. 2312, 2020.

[6] T. Falk et al., "U-net: deep learning for cell counting, detection, and morphometry," *Nature Methods*, vol. 16, no. 1, pp. 67–70, 2019.

[7] U. Schmidt, M. Weigert, C. Broaddus, and G. Myers, "Cell detection with star-convex polygons," in *Proceedings of MICCAI'18*, Granada, Spain, September 16-20, 2018, pp. 265–273.

[8] P. Brigger, J. Hoeg, and M. Unser, "B-Spline snakes: A flexible tool for parametric contour detection," *IEEE Transactions on Image Processing*, vol. 9, no. 9, pp. 1484–1496, 2000.

[9] R. Delgado-Gonzalo, P. Thévenaz, C.S. Seelamantula, and M. Unser, "Snakes with an ellipse-reproducing property," *IEEE Transactions on Image Processing*, vol. 21, no. 3, pp. 1258–1271, 2012.

[10] V. Uhlmann, J. Fageot, and M. Unser, "Hermite snakes with control of tangents," *IEEE Transactions on Image Processing*, vol. 25, no. 6, pp. 2803–2816, 2016.

[11] A. Badoual, D. Schmitter, V. Uhlmann, and M. Unser, "Multiresolution subdivision snakes," *IEEE Transactions on Image Processing*, vol. 26, no. 3, pp. 1188–1201, 2017.

[12] A. Song, V. Uhlmann, J. Fageot, and M. Unser, "Dictionary learning for two-dimensional kendall shapes," *SIAM Journal on Imaging Sciences*, vol. 13, no. 1, pp. 141–175, 2020.

[13] S. Mandal and V. Uhlmann, "A learning-based formulation of parametric curve fitting for bioimage analysis," in *Proceedings of ENUMATH'19*, Egmond aan Zee, The Netherlands, September 30 - October 4, 2019.

[14] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*, MIT Press, Cambridge, MA, USA, 2016.

[15] M. Unser, "Splines: A perfect fit for signal and image processing," *IEEE Signal Processing Magazine*, vol. 16, no. 6, pp. 22–38, 1999.

[16] S. Satoshi and K. Abe, "Topological structural analysis of digitized binary images by border following," *Comput Gr Image Process*, vol. 30, no. 1, pp. 32 – 46, 1985.

[17] M. Abadi et al., "Tensorflow: A system for large-scale machine learning," in *Proceedings of OSDI'16*, Savannah, GA, USA, November 2–4, 2016, pp. 265–283.

[18] J. Caicedo et al., "Nucleus segmentation across imaging experiments: the 2018 data science bowl," *Nature Methods*, vol. 16, no. 12, pp. 1247–1253, 2019.

[19] T. Blu and M. Unser, "Quantitative Fourier analysis of approximation techniques: Part I—Interpolators and projectors," *IEEE Trans Signal Process*, vol. 47, no. 10, pp. 2783–2795, 1999.

[20] M. Levandowsky and D. Winter, "Distance between sets," *Nature*, vol. 234, no. 5323, pp. 34–35, 1971.

[21] M. Unser, "Sampling—50 Years after Shannon," *Proceedings of the IEEE*, vol. 88, no. 4, pp. 569–587, 2000.