

# A comprehensive review and new taxonomy on superpixel segmentation

ISABELA BORLIDO BARCELOS, Pontifical Catholic University of Minas Gerais, Brazil

FELIPE DE CASTRO BELÉM, University of Campinas, Brazil

LEONARDO DE MELO JOÃO, University of Campinas, Brazil

ZENILTON K. G. DO PATROCÍNIO JR., Pontifical Catholic University of Minas Gerais, Brazil

ALEXANDRE XAVIER FALCÃO, University of Campinas, Brazil

SILVIO JAMIL FERZOLI GUIMARÃES, Pontifical Catholic University of Minas Gerais, Brazil

Superpixel segmentation consists of partitioning images into regions composed of similar and connected pixels. Its methods have been widely used in many computer vision applications since it allows for reducing the workload, removing redundant information, and preserving regions with meaningful features. Due to the rapid progress in this area, the literature fails to catch up on more recent works among the compared ones and to categorize the methods according to all existing strategies. This work fills this gap by presenting a comprehensive review with new taxonomy for superpixel segmentation, in which methods are classified according to their processing steps and processing levels of image features. We revisit the recent and popular literature according to our taxonomy and evaluate 20 strategies based on nine criteria: connectivity, compactness, delineation, control over the number of superpixels, color homogeneity, robustness, running time, stability, and visual quality. Our experiments show the trends of each approach in pixel clustering and discuss individual trade-offs. Finally, we provide a new benchmark for superpixel assessment, available at <https://github.com/IMScience-PPGINF-PucMinas/superpixel-benchmark>.

**CCS Concepts:** • General and reference → Surveys and overviews; • Computing methodologies → Image segmentation.

**Additional Key Words and Phrases:** superpixel, image segmentation, survey, image processing

## ACM Reference Format:

Isabela Borlido Barcelos, Felipe de Castro Belém, Leonardo de Melo João, Zenilton K. G. do Patrocínio Jr., Alexandre Xavier Falcão, and Silvio Jamil Ferzoli Guimarães. 2024. A comprehensive review and new taxonomy on superpixel segmentation. *ACM Comput. Surv.* 1, 1 (October 2024), 54 pages. <https://doi.org/10.1145/3652509>

## 1 INTRODUCTION

Superpixel segmentation aims to divide images into homogeneous regions of connected pixels, such that unions of superpixels compose image objects. It has several benefits, such as reducing the workload (e.g., reducing millions of pixels to thousands/hundreds of superpixels) and providing higher-level content information than pixels. Consequently, methods for superpixel segmentation are used in several applications, such as object segmentation [18, 68, 104], anomaly detection [93] semantic segmentation [145], saliency detection [143, 146], and image classification [35, 100].

---

Authors' addresses: Isabela Borlido Barcelos, isabela\_borlido@hotmail.com, Pontifical Catholic University of Minas Gerais, Belo Horizonte, Minas Gerais, Brazil, 30535-901; Felipe de Castro Belém, felipe.belem@ic.unicamp.br, University of Campinas, Campinas, Brazil, 13083-970; Leonardo de Melo João, leonardo.joao@ic.unicamp.br, University of Campinas, Campinas, Brazil, 13083-970; Zenilton K. G. do Patrocínio Jr., zenilton@pucminas.br, Pontifical Catholic University of Minas Gerais, Belo Horizonte, Minas Gerais, Brazil, 30535-901; Alexandre Xavier Falcão, afalcao@ic.unicamp.br, University of Campinas, Campinas, Brazil, 13083-970; Silvio Jamil Ferzoli Guimarães, sjamil@pucminas.br, Pontifical Catholic University of Minas Gerais, Belo Horizonte, Minas Gerais, Brazil, 30535-901.

---

© 2024 Copyright held by the owner/author(s). Publication rights licensed to ACM.

This is the author's version of the work. It is posted here for your personal use. Not for redistribution. The definitive Version of Record was published in *ACM Computing Surveys*, <https://doi.org/10.1145/3652509>.

Superpixel segmentation has a vast literature, and although previous work provided categorizations [1, 61, 109] and benchmarks [82, 87, 109, 122] to evaluate and compare methods, such works did not cover more recent approaches. Figure 1 presents three superpixel segmentation examples, in which the superpixels' borders are shown in red. In the literature, several authors identified the desired superpixel properties. Despite the absence of consensus, most authors agreed that superpixels must be composed of connected pixels, adhere to the objects' borders, present smooth contours, and have regularly distributed and compact shapes [109, 122]. Moreover, the methods must be computationally efficient and generate a controllable number of superpixels. However, superpixel methods usually meet part of those criteria, which often occurs when the improvement in a property leads to worse for another property. For instance, Figure 1(a) has superpixels with maximum compacity and regularity, but their contours do not adhere to the object's borders. Improving boundary adherence may negatively impact compactness (Figure 1(c)). Some superpixel approaches try to manage this trade-off (Figure 1(b)). In this sense, the choice of an evaluation measure depends on the optimized property.

In contrast to the rapid progress in new superpixel strategies, the papers usually compared their proposals against classical approaches. Therefore, there are few comparisons among state-of-the-art methods, which impairs the judgment of their actual contribution. Benchmarks usually fill this gap by offering an easy-to-use tool to compare different approaches. The first benchmark for superpixel evaluation [87] compared eight algorithms and evaluated object delineation and robustness to affine transformations. To overcome the biased penalty in *Under-segmentation Error* (UE) measure [64] caused by the superpixel size, the authors proposed a modified UE to consider the smallest part of the superpixel leakage. Also, the evaluated superpixel methods presented similar results, demonstrating that the most appropriate methods for each task depend on the crucial characteristics of that task. In addition, algorithms less focused on compactness showed greater robustness to image transformations. Unlike Neubert and Protzel [87], Achanta et al. [1] demonstrated the effectiveness of *Simple Linear and Iterative Clustering* (SLIC) by comparing five superpixel methods to determine their benefits and limitations regarding their boundary adherence and efficiency. Achanta et al. [1] characterized the superpixel methods as *graph-based* and *gradient-ascent-based*. The former contains methods that model the segmentation problem based on graph theory generating superpixels by minimizing a cost function defined on the graph. The second iteratively refines its initial clusters until reaching a convergence criterion. Although the categorization provided [1] is widely adopted in the literature on superpixels, it fails to cover recent strategies.

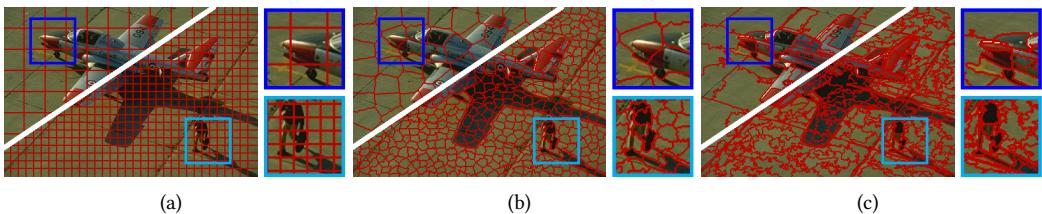


Fig. 1. Superpixel segmentation examples, in which superpixel borders are shown in red. Although boundary adherence, regularity, and compactness are essential properties, (a) superpixels with higher regularity and compactness have poor boundary adherence. Conversely, (b) superpixel methods focused on boundary adherence may present irregular contours due to their sensitivity to subtle color variations.

Schick et al. [98, 99] investigated the importance of compactness in superpixel segmentation. They proposed a compactness measure based on the isoperimetric coefficient [90] and demonstrated a trade-off between Compactness and Boundary Recall [81]. The authors argued that a more accurate segmentation would not imply better overall performance. Thus, they claimed that compact superpixels better capture spatially coherent information facilitating information extraction from their boundaries. In contrast, Stutz et al. [108] explored the impact of depth information in superpixel methods in a benchmark with fifteen algorithms and two datasets. According to their evaluation, depth inclusion may not represent improved results. Regarding visual quality, the authors settled that the high quantitative results in the delineation assessment did not necessarily reflect the segmentations' visual quality. Mathieu et al. [82] argued that more than two datasets, as used in [108], are needed for an exhaustive evaluation. They overcome this with a new dataset, called the *Heterogeneous Size Image Dataset* (HSID). The HSID mainly contains large images (with millions of pixels) and allows evaluating the superpixel methods according to the image size. Using the HSID, the authors analyzed the five best superpixel methods in [108] and Waterpixels [77] method. The evaluated methods did not achieve a satisfactory trade-off between adherence to contours, conciseness (smallest possible number of superpixels), and efficiency. Therefore, the authors argued that the superpixel method must be chosen according to the necessary superpixels' characteristics for the desired task.

Wang et al. [122] proposed a regularity measure for superpixels, allowing a quantitative regularity analysis. The authors also provided an overview of the superpixel methods and a benchmark with fifteen methods and thirteen evaluation measures, including the proposed one. In [122], the superpixel methods were categorized as clustering-based (or gradient-based) and graph-based, following the characterization in [1]. According to Wang et al. [122], methods based on clustering showed greater efficiency, while those based on graphs presented an improved delineation. However, the authors argued that the evaluated algorithms are hardly applicable in scenarios requiring real-time responses. The authors in [109] presented a more comprehensive evaluation in a benchmark with 28 superpixel algorithms with five datasets that included indoor, outdoor, and people images. In addition to the benchmark, the authors also proposed three evaluation measures independent of the number of superpixels and based on existing delineation metrics: *Average Miss Rate* (AMR), *Average Under-segmentation Error* (AUE), and *Average Unexplained Variation* (AVU). Stutz et al. [109] evaluated the stability of superpixel methods, considering the minimum, maximum, and standard deviation of each metric; and its robustness to noise, blur, and affine transformations. Based on the categorization in [1], they also categorized superpixel methods by their high-level approach, allowing them to relate their categories to experimental results. Despite the broad categorization in [109], the authors settled that some methods in the literature are not included in their categorization. Based on the proposed evaluation, they created a ranking of the evaluated methods, in which they recommended six of them: ETSPS [135], SEEDS [114], ERS [71], CRS [28], ERGC [20], and SLIC [1].

Recently, the authors in [61] extensively discussed various aspects of superpixel segmentation. They reviewed several classical superpixel methods and categorized them as *graph-based*, *clustering-based*, *watershed-based*, *energy optimization*, and *wavelet-based* techniques. They also reviewed superpixel methods based on the classical approaches, extensively discussed them for general purposes and specific domains, and presented some commonly used datasets and evaluation measures. However, the work in [61] did not perform an experimental evaluation. Although the desired attributes of superpixels were broadly discussed, which methods are more advantageous than others are still to be determined.

Other recent works discussed superpixel segmentation for specific applications, such as superpixels as pre-processing for clustering [97] and superpixels in hyperspectral images [46]. Clustering and superpixel methods categorizations were also provided in [97], where superpixel methods were

categorized as *density-based*, *watershed-based*, *graph-based*, *path-based*, *contour evolution-based*, *energy optimization-based*, and *clustering-based* methods. In [97], the authors evaluated the efficacy of combining superpixels and partitional clustering approaches using SLIC as pre-processing in rosette plant images and oral histopathology images. Their results indicated that although the pre-processing based on superpixels could not improve accuracy, it reduced execution time and produced more compact, coherent, and regular image regions.

Despite the evaluations in previous benchmarks, superpixel approaches have made significant progress in recent years by introducing new strategies, making previous reviews and evaluations outdated. This work presents an overview of several superpixel segmentation strategies from both classic and recent literature. We also introduce a new benchmark that includes six superpixel methods recommended by [109] and seventeen recent algorithms. Moreover, we provide a comprehensive assessment based on nine well-established criteria: *delineation*; *compactness*; *color homogeneity*; *running time*; *connectivity*; *control over the number of superpixels*; *robustness*; *stability*; and *visual quality*. The results provide valuable insights into the pros and cons of the methods, supporting the choice of the most suitable one for a given application.

This paper is organized as follows. Section 2 describes the proposed taxonomy and categorizes the most recent and commonly used superpixel methods. Section 3 presents the benchmark setup, including methods, datasets, and evaluation criteria. Section 4 presents the obtained results in five datasets and 23 superpixel methods. Finally, we draw conclusions and state future work in Section 5. In addition, we provide supplementary material with three appendices. The reader should refer to Appendix A for an extensive description covering several superpixel methods. In terms of evaluation, quantitative benchmark measures are presented in Appendix B. Furthermore, Appendix C provides additional results with experiments evaluating connectivity, stability, and robustness, along with a review of the overall performance concerning the clustering categories.

## 2 TAXONOMY OF SUPERPIXEL METHODS

Most articles categorize superpixel methods into clustering-based, graph-based, and, more recently, deep-learning proposals. Only a few recent works [61, 109] present more categories for such methods. However, while the categories in [109] cannot represent some recent superpixel methods, the authors in [61] focused mainly on classical approaches. A taxonomy based on different and non-strict aspects may be more appropriate since previous categorizations do not cover the wide variety of superpixel approaches, and the rapid advance in this area hampers the establishment of disjoint categories. Therefore, this work provides a taxonomy that categorizes methods according to their processing steps and the abstraction level of the features used. In addition, it also reports the desired superpixel properties that each method satisfies.

### 2.1 Processing steps

To provide a comprehensive taxonomy with a more natural representation, we identified that superpixel algorithms generally have up to three steps: (i) initial; (ii) main; and (iii) final processing. We identify categories that broadly define the process performed at each processing step in 59 superpixel segmentation methods. Figure 2 provides an overview of the categories for each processing step. For instance, in *Initial Processing*, superpixel methods usually perform image pre-processing, such as denoising or feature extraction, or the methods compute the initial algorithm setup, such as creating seeds or performing an initial segmentation. On the other hand, the *Main Processing* step contains the strategy for superpixel computation, including the whole loop for superpixel generation, if any. As one may see in Figure 2, some main processing categories have deep networks, which we categorize based on the pixel-superpixel assignment process and the network's output. After computing superpixels, post-processing operations (the *Final Processing*)

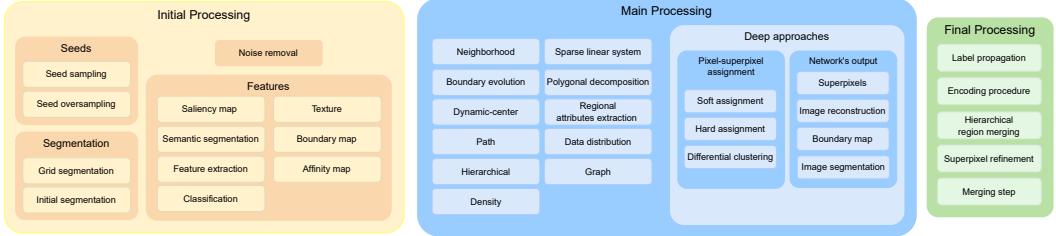


Fig. 2. Categories of each processing step in superpixel taxonomy.

may ensure superpixel connectivity, fine-tune the segmentation, or complete the pixel-superpixel map computation.

The processing steps in our taxonomy divide superpixel approaches into specialized procedures, from which one may identify categories. Table 1 presents the categories of the *Main Processing* step whose clustering procedure does not use convolutional networks. Our taxonomy introduces some new categories and also reviews others. Instead of the common *clustering-based* (also called gradient-based), our taxonomy contains the *neighborhood-based* and *dynamic-center-update* clustering categories. The former performs clustering restricted to a maximum spatial distance from some reference point in the image, while the latter dynamically updates the cluster centers based on an optimization function. Furthermore, the *graph-based* category here relates to using graph topology instead of graph modeling. Also, similar to Stutz et al. [109], our taxonomy includes *boundary evolution*, *path-based*, and *density-based* clustering categories. Finally, we introduce the categories *sparse linear system*, *data distribution-based*, *regional feature extraction*, *polygonal decomposition*, and *hierarchical* clustering. Table 1 shows their definitions.

## 2.2 Processing level of image features

Superpixel methods can either compute features on the fly or obtain them from other algorithms. Additionally, several approaches combine the same information differently to extract features. For instance, some methods combine local features (e.g., color and pixel position) with higher-level ones (e.g., edge or semantic information) in their optimization function [15, 125, 142]. Conversely, other extracted features by only exploring local information – e.g., using strategies based on graph theory or linear algebra [14, 23, 38]. However, as far as we know, there was no study on the features' impact on superpixel generation. Although such a study is beyond the scope of this work, we categorize superpixel methods based on the processing level of the features used. Since superpixel methods usually combine higher-level features with lower-level ones, we categorize them according to the highest-level features. The categories are defined as follows:

- **Pixel-level features:** raw data resources in images – e.g., pixel color, position, and depth;
- **Mid-level features:** features that can be computed based on a set of pixels, smaller than the entire image – e.g., patch-based feature, path-based feature, gradient, or boundary;
- **High-level features:** features that combine pixel properties and high-level information. The high-level information cannot be extracted from a small set of pixels. They are given directly by the user or predicted by other models – e.g., saliency map, semantic features, texture, or a desired object geometry.

## 2.3 The proposed taxonomy in superpixel literature

This section presents our taxonomy applied to superpixel literature, in which we categorize the processing steps of **59 superpixel methods**. In the following, we discuss the main processing

Table 1. Main processing categories excluding those based on neural networks.

Clustering categories	Explanation
Neighborhood-based	Performs clustering based on the similarity between pixels restricted to a maximum spatial distance from some reference point in the image.
Boundary evolution	These algorithms iteratively update the superpixels' boundaries to optimize an energy function, usually using a coarse-to-fine image block strategy.
Dynamic-center-update	The dynamic-center-update algorithms perform clustering with a distance function based on the features of the clusters, dynamically updating their centers.
Path-based	Path-based approaches generate superpixels by creating paths in the image graph based on some criteria. Usually, its clustering criterion is a path-based function to optimize during clustering.
Hierarchical	These algorithms create regions in the image that form a hierarchical structure, obeying the criteria of locality and causality [50].
Density-based	These superpixel methods model the problem of computing superpixels in a problem of finding density peaks.
Sparse linear system	Model the segmentation problem with a sparse matrix and use its properties to find superpixels.
Data distribution-based	The approach assumes that the image pixels follow a specific distribution and perform the clustering based on this conjecture.
Regional feature extraction	Iteratively extracts regional features to perform clustering based on these features.
Polygonal decomposition	The segmentation in these methods consists of decomposing the image into non-overlapping polygons.
Graph-based	Perform superpixel segmentation based on graph topology.

categories and the usage of deep learning in superpixel segmentation. Then, we present the complete taxonomy applied to the superpixel methods.

Figure 3 summarizes the superpixel methods according to their main processing categories. The **neighborhood-based** methods usually require an initial seed sampling, in which seeds represent superpixel centers, and a final merging step ensures connectivity since their neighborhood distance usually allows superpixels to conquer non-connected pixels [1, 23, 41, 51, 69, 125, 129, 142]. Also, most neighborhood-based methods manage compactness by parameter. In contrast, methods with **boundary evolution-based** clustering require an initial segmentation, but they usually guarantee connectivity since only pixels at superpixels' borders can conquer neighbor pixels [17, 28, 67, 88, 91, 114, 131, 135, 139]. Their initial grid segmentation and the restricted pixel-conquering strategy allow the creation of highly compact and regular superpixels, while the iterative coarse-to-fine block strategy improves delineation. Boundary evolution-based methods are usually more efficient than other approaches, although they usually do not produce the precise number of superpixels.

In **dynamic-center-update** algorithms, the optimization function usually relies on the superpixel centers' features, dynamically updating them to improve these features [2, 43, 57, 66, 74, 123, 144]. Most of these methods avoid performing several iterations, updating each pixel once with a priority queue. They usually have good boundary adherence but less compactness than neighborhood-based and boundary evolution-based clustering methods. In contrast, superpixel methods with **path-based** clustering are usually focused on delineation rather than compactness [13–15, 21, 116]. Similar to neighborhood-based methods, they require an initial seed sampling, but instead of superpixel centers, the seeds are the roots of the trees. In path-based strategies, superpixels are usually trees that start with a unique seed and iteratively conquer pixels according to the graph's adjacency. Such a clustering procedure allows the development of optimization functions based on the paths (tree branches) instead of a global function, and the pixel conquering based on the graph's adjacency guarantees connectivity. Methods with **hierarchical** clustering create a

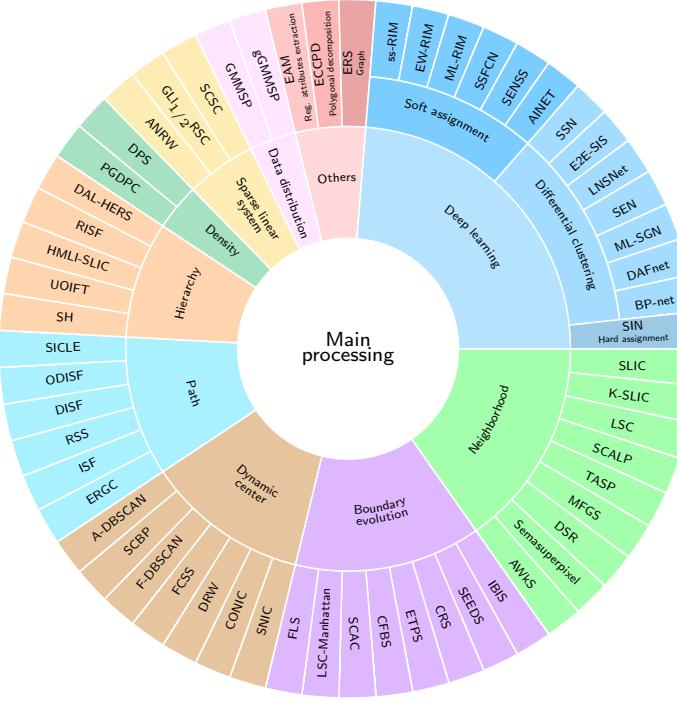


Fig. 3. The main processing categories in superpixel taxonomy and the methods that conform with each one.

hierarchical structure by iteratively merging pixels or dividing image regions [9, 30, 38, 89, 127]. Instead of computing only a pre-determined number of superpixels, most of these methods extract different superpixel quantities, named scales, from the hierarchical structure. However, to improve running time and delineation, the hierarchical structure can have dense and sparse scales and, therefore, they may not produce any superpixel quantity. Similar to path-based clustering methods, the hierarchical ones usually focus on boundary adherence, and their strategy to cluster guarantees connected superpixels. Although they require a unique execution to produce all scales, the superpixel leakage at one hierarchical scale is propagated to the following ones, increasing delineation error.

**Density-based** methods model the problem of finding superpixels in the problem of finding density peak pixels [47, 102]. Similar to path-based and hierarchical-based methods, the density-based ones also focus on delineation, but they may not guarantee connectivity. Also, unlike most neighborhood-based and boundary evolution-based methods, density-based methods usually use non-iterative approaches and they assume that the image pixel features form peaks of density (groups of similar pixels) along the image dimension, considering them density peaks as candidates for superpixel centers. Similarly, **data distribution-based** approaches assume that features in image pixels follow a specific distribution. In this work, only GMMSP [6] performs such a strategy and considers that the image pixels follow a Gaussian distribution. GMMSP does not allow direct control over the number of superpixels and does not produce highly compact superpixels. However, its superpixels have smooth borders and low variation in size. In contrast, **sparse linear system clustering** methods model pixel similarities with a sparse matrix, using algorithms based on linear

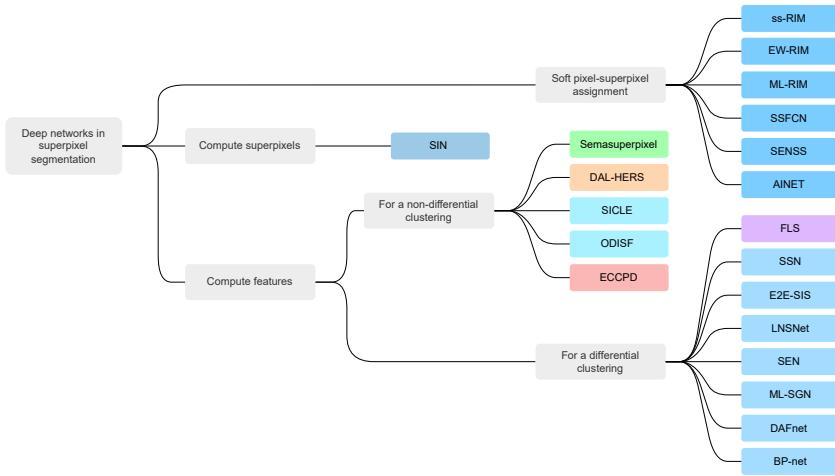


Fig. 4. The usage of neural networks in superpixel segmentation. The color of each method relates to its main processing category color in Figure 3.

algebra to solve the segmentation problem [36, 65, 118]. These methods usually have a higher time complexity, prioritize delineation over homogeneity, and may not guarantee connectivity.

**Regional feature extraction clustering** methods iteratively extract features from image regions and use these features to perform clustering. EAM [4], the unique superpixel method in this work with this clustering approach, performs an iterative coarse-to-fine grid segmentation based on attributes extracted from image regions. Although such a procedure is similar to boundary evolution clustering, EAM does not improve its superpixels during the iterative process. Instead, it performs a further merging stage to compose superpixels. Unlike boundary evolution clustering methods, EAM does not generate compact or regular superpixels. However, it can capture finer details by producing fewer superpixels in homogeneous regions. On the other hand, **polygonal decomposition clustering** methods decompose the image into non-overlapping polygons as superpixels. In this work, only ECCPD [76] uses this clustering strategy. The ECCPD has highly compact and connected superpixels compared to other clustering methods. However, it requires minutes to segment an image. The **graph-based** clustering performs superpixel segmentation based on graph topology. In this work, only ERS [71] uses this clustering strategy. In contrast to ECCPD, ERS uses an efficient greedy algorithm to solve the problem of selecting a set of edges to find a predetermined number of connected components in a graph. ERS has a balancing term to control compactness, and the graph's adjacency guarantees connectivity. Also, ERS can generate the exact number of desired superpixels.

The remaining clustering categories relate to **deep-learning networks**. The neural networks used for superpixels are typically deep convolutional, and they are used in the main or initial processing. Although deep learning is a popular topic in computer vision, its use for superpixel segmentation is relatively new. This delay is due to two major challenges: (i) propose differentiable operations for the pixel-superpixel association and (ii) fit the irregular superpixel lattices into regular convolutional ones. As a result, most deep-learning networks do not produce superpixels directly. Instead, they usually employ a differential clustering module in an end-to-end trainable network. As shown in Figure 4, superpixel segmentation methods may use deep-learning networks to (i) extract features for a non-differential clustering module, (ii) compute pixel-superpixel soft association using a differential clustering module, (iii) compute pixel-superpixel soft association

directly, or (iv) compute superpixels directly. The deep networks in (i) perform an initial processing for a further clustering step. Conversely, in (ii), the network's training procedure usually integrates a differential clustering module, and, in this case, we consider that the network performs clustering. In this work, only FLS [88] adopts a differential clustering module without integrating it into the network's training process. Finally, the deep networks in (iii) and (iv) also perform clustering and, therefore, are part of the main processing step.

The SSN [55] overcomes these issues with a supervised fully convolutional network to extract image features and a differentiable clustering module based on SLIC [1] to produce a pixel-superpixel soft association. As far as we know, the proposal in [55] was the first end-to-end trainable network for superpixel segmentation and inspired others. For instance, E2E-SIS [120], BP-net [141], and DAFnet [130] are supervised deep-based superpixel approaches that also use a differentiable clustering module based on SLIC. E2E-SIS performs multi-task learning that exploits the mutual benefit between image segmentation and superpixel segmentation. In contrast, BP-net and DAFnet generate superpixels for RGB-D and stereo images, respectively. Other approaches employ new clustering modules, such as SEN [39] with a differential mean-shift module and LNSNet [147] with a Non-iterative Clustering Module. Both are unsupervised networks, in which the former uses superpixels generated from SNIC [2] as pseudo-ground-truth, and the latter adopts a lifelong learning strategy. Similarly, SSFCN [134] and ML-SGN [70] use a U-shaped network, in which the former employs a supervised strategy that directly outputs a pixel-superpixel association map, and the latter uses an unsupervised strategy with a differential clustering based on SLIC to train a multitasking network. Inspired by SSFCN, SENSS [119], and AINET [126] are also supervised U-shaped networks, in which the former improves learning ability with *Squeeze-and-Excitation* blocks, and the latter employs a boundary-perceiving loss to improve boundary delineation and an *Association Implantation* module to associate each pixel with its surrounding superpixels in a grid shape. Conversely, some deep-learning methods integrate the soft pixel-superpixel assignment into the convolutional process. For instance, ss-RIM [110], EW-RIM [136], and ML-RIM [32] use the deep image prior procedure [63] to generate superpixels without image ground truth. Instead, they are trained based on clustering entropy, spatial smoothness, and reconstruction.

The deep learning-based approaches are all end-to-end trainable. However, the aforementioned deep-based methods train soft pixel-superpixel assignments, requiring a post-processing step to compute hard associations. The interpolation network in SIN [138] overcomes it by extracting features with convolutional operations followed by multiple interpolations to expand the pixel-superpixel association matrix while enforcing spatial connectivity. Table 2 presents superpixel methods according to the proposed taxonomy, their superpixel properties (second to third columns), color space, time complexity (when available), and inspiration method (if any). However, instead of indicating the pixel-superpixel assignment category in the *Main Processing* of deep learning methods (as in Figure 3), we complement it by informing, along with the network output (*out*), its architecture (*arch*) for methods with CNN in any processing step. As far as we know, there is no categorization for deep convolutional networks. Therefore, we classify each architecture according to its most important aspect. In Table 2, the superpixel properties are whether a method is iterative (Iterative), its control over the number of iterations (#*Iter.*) and the number of superpixels (#*Superp.*), whether its superpixels are connected (*Connec.*) and compact (*Compact.*), and if the network training (if any) is supervised (*Superv.*). One may note that a method may perform several procedures in a processing step, implying that more than one category may appear. For instance, DSR, Semasuperpixel, ODISF, SICLE, EAM, and ECCPD have two categories each in the initial processing, since each one performs two distinct processes before the main processing (*i.e.*, before the clustering strategy). The reader should refer to Appendix A for a detailed description of each method in Table 2.

Table 2. Recent methods for superpixel segmentation.

Method	Iterative	aIter.	aSuperp.	aCompct.	aColor	Time complexity	Initial processing	Main processing	Final processing	Features			
										Pix.	Mid.	High	
SLIC [1]	✓	✓	✓	✓ <sup>a</sup>	✓	CIELAB	Seed sampling	Neighborhood-based	Merging step	✓			
K-SLIC [113]	✓	✓	✓	✓ <sup>a</sup>	✓	RGB	Compute optimum K	Neighborhood-based	Merging step	✓			
LSC [67]	✓	✓	✓	✓ <sup>a</sup>	✓	CIELAB	Seed sampling	Neighborhood-based	Merging step	✓		SLIC [1]	
SCALP [41]	✓	✓	✓	✓	✓	CIELAB	Seed sampling	Neighborhood-based		✓		SLIC [1]	
TASP [129]	✓	✓	✓	✓	✓	CIELAB	Seed sampling	Neighborhood-based	Merging step	✓		SLICO [1]	
MFGS [69]			✓ <sup>c</sup>	✓	✓	CIELAB	Seed sampling	Neighborhood-based	Merging step	✓			
DSR [142]	✓		✓ <sup>c</sup>	✓	✓	CIELAB	Saliency computation and Seed sampling arch: Encoder-decoder	Neighborhood-based	Merging step	✓		dSLIC [78]	
Semasuperpixel [125]													
AWS [51]	✓	✓	✓	✓	✓	CIELAB	Seed sampling	Neighborhood-based	Merging step	✓			
IBIS, IBSSouls [17]	✓	✓	✓	✓ <sup>a</sup>	✓	CIELAB	O( $kn + nz$ ) <sup>b</sup>	Boundary evolution	Merging step	✓			
SEEDS [114, 115]	✓	✓	✓	✓	✓	CIELAB	Grid segmentation	Boundary evolution	Merging step	✓			
CRCS [28]	✓	✓	✓	✓	✓	YCbCr	Grid segmentation	Boundary evolution		✓		CR [48, 83]	
ETPS [135]	✓	✓	✓	✓	✓	RGB	Grid segmentation	Boundary evolution		✓		SEEDS [114]	
CFBS [131]	✓	✓	✓	✓	✓	CIELAB	Grid segmentation	Boundary evolution		✓		SLIC [1]	
SCAC [139]			✓ <sup>c</sup>	✓	✓	CIELAB	Grid segmentation	Boundary evolution	clustering	✓		WSBM [140]	
LSC-Manhattan [91]													
FLS [68]	✓	✓	✓	✓	✓	CIELAB	Texture complexity classification arch: FCN	Boundary evolution				LSC [23]	
SNIC [2]			✓	✓	✓	CIELAB	O( $n$ )	out: Affinity map			✓		
CONIC [48]			✓	✓	✓	CIELAB	O( $n$ )	Seed sampling	Dynamic-center-update	✓		SEEDS [115]	
DRW [77]			✓	✓	✓	CIELAB	O( $n$ )	Seed sampling	Dynamic-center-update	✓		SLIC [1]	
FCSS [66]	✓	✓ <sup>c</sup>	✓	✓ <sup>a</sup>	✓	CIELAB	O( $(n + nt)$ ) <sup>d</sup>	Seed sampling	Label propagation	✓		SNIC [2]	
F-DBSCAN [74]			✓	✓	✓	CIELAB	O( $n$ )	Seed sampling	Dynamic-center-update	✓		RT-DBSCAN [44]	
SCB [144]			✓	✓	✓	RGB	O( $n$ )	Seed sampling	Dynamic-center-update	✓		DBSCAN [103]	
A-DBSCAN [123]			✓	✓	✓	RGB	O( $n$ )	Seed sampling	Dynamic-center-update	✓		DBSCAN [103]	
ERGC [20]			✓	✓	✓	CIELAB	O( $n$ )	Seed sampling	Path-based	✓			
ISF [116]	✓	✓	✓	✓	✓	CIELAB	O( $n \log n$ )	Seed sampling	Path-based	✓		IFT [34]	
RSS [21]			✓	✓	✓	CIELAB	O( $n$ )	Seed sampling	Path-based	✓		IFT [34]	
DISF [14]	✓		✓	✓	✓	CIELAB	O( $n \log n$ )	Seed oversampling arch: Encoder-decoder	Path-based	✓		ISF [116]	
ODISF [15]	✓	✓	✓	✓	✓	CIELAB	O( $n \log n$ ) <sup>e</sup>	out: Saliency map and Seed oversampling arch: Encoder-decoder	Path-based	✓		DISF [14], OISF [12]	
SICLE [11, 13]													
SH [127]			✓	✓	✓	RGB	O( $n$ )	Seed sampling and Seed oversampling	Path-based	✓		ODISF [15]	
UOIFT [9]			✓	✓	✓	CIELAB	Hierarchical	Clustering method	Hierarchical	✓			
HMLI-SLIC [30]	✓	✓	✓ <sup>c</sup>	✓	✓	CIELAB	O( $nd$ ) <sup>f</sup>	Clustering method	Hierarchical	✓		IIFT [34], OIFT [79]	
RISF [37, 38]	✓	✓	✓	✓	✓	CIELAB	Hierarchical	Region merging	Hierarchical	✓		SLIC [1]	
DAL-HERS [89]			✓	✓	✓	RGB	O( $n$ ) <sup>g</sup>	arch: Multi-scale Residual CNN	Hierarchical	✓		SEAL [112], ERS [71]	
PGDPC [47]			✓	✓	✓	CIELAB	O( $n \log n$ )	out: Affinity map				DPC [117]	
DPS [102]			✓ <sup>c</sup>	✓	✓	CIELAB	O( $n \log n$ )	Seed sampling	Density-based	✓		DP [95]	
ANRW [118]			✓	✓	✓	YCbCr	O( $n^2$ )	Compute features	Density-based	✓		NRW [137]	
GLI <sub>1/2</sub> RSC [36]	✓		✓	✓	✓	RGB	Hierarchical	Clustering method	Sparsify system	✓		CAWR [124]	
SCSC [65]	✓	✓	✓	✓	✓	RGB	Hierarchical	Clustering method	Sparsify system	✓			
EAM [4]			✓ <sup>c</sup>	✓	✓	RGB	O( $\log^2 n$ )	Noise removal and Boundary map computation arch: Multi-scale CNN	Regional attributes extraction	✓			
ECCPD [76]	✓	✓	✓	✓	✓	RGB	Boundary map and Seed sampling	Polygonal decomposition	Boundary evolution clustering	✓			
GMMSNP [6]	✓	✓	✓ <sup>c</sup>	✓ <sup>a</sup>	✓	CIELAB	O( $n$ ) <sup>h</sup>	Data distribution-based	Merging Step	✓		SCGAGMM [56], GMMSNP [6]	
gGMMSNP [7]	✓	✓	✓ <sup>c</sup>	✓ <sup>a</sup>	✓	CIELAB	O( $n$ ) <sup>h</sup>	Data distribution-based	Merging Step	✓			
ERS [71]			✓	✓	✓	RGB	Hierarchical	Graph-based arch: FCN	Merging Step	✓			
SSN [55]			✓ <sup>c</sup>	✓ <sup>a</sup>	✓	CIELAB	Hierarchical	out: Superpixels arch: FCN	Merging Step	✓		SLIC [1]	
E2E-SIS [120]			✓	✓ <sup>a</sup>	✓	CIELAB	Hierarchical	out: Superpixels and image segmentation arch: FCN	Merging Step	✓		DEL [73], SSN [55]	
LNS-net [147]			✓	✓	✓	LAB/RGB	Hierarchical	Image reconstruction and Superpixels arch: Encoder-Decoder	Merging step	✓			
ss-RIM [110]			✓ <sup>c</sup>	✓	✓	RGB	Hierarchical	out: Image reconstruction and Superpixels arch: Encoder-Decoder	Merging step	✓		DIP [63], RIM [60]	
EW-RIM [136]			✓ <sup>c</sup>	✓	✓	RGB	Hierarchical	out: Image reconstruction and Superpixels arch: Encoder-Decoder	Merging step	✓		ss-RIM [110], DIP [63]	
ML-RIM [32]			✓ <sup>c</sup>	✓	✓	RGB	Hierarchical	arch: Encoder-Decoder multi-scale module	Image reconstruction and Superpixels	✓		ss-RIM [110], EW-RIM [136]	
SEN [39]			✓	✓	✓	RGB	Hierarchical	arch: Encoder-Decoder	out: Superpixels	✓		RPEIG [59]	
ML-SGN [70]			✓ <sup>c</sup>	✓ <sup>a</sup>	✓	RGB	Hierarchical	arch: Encoder-Decoder	out: Superpixels and image segmentation	✓		SSN [55]	
SSFCN [134]			✓ <sup>c</sup>	✓ <sup>a</sup>	✓	CIELAB	Hierarchical	arch: Encoder-Decoder	out: Superpixels	✓		SSN [55]	
SENSS [119]			✓ <sup>c</sup>	✓	✓	CIELAB	Hierarchical	arch: Encoder-Decoder	out: Superpixels	✓		SSFCN [134]	
AINET [126]			✓ <sup>c</sup>	✓ <sup>a</sup>	✓	CIELAB	Hierarchical	arch: Encoder-Decoder	out: Superpixels	✓		SSFCN [134]	
DAFnet [130]			✓	✓	✓	CIELAB	Hierarchical	arch: Weight-shared CNN	out: Superpixels	✓		SSFCN [134]	
SIN [138]			✓ <sup>c</sup>	✓	✓	CIELAB	Hierarchical	arch: Interpolation Network	out: Superpixels	✓			
BP-net [141]			✓	✓	✓	RGB-D	Seed sampling	arch: Multi-scale CNN and FCN	out: Boundary map and superpixels	Merging step	✓		

<sup>a</sup> With post-processing. <sup>b</sup>  $k$  is the number of iterations and  $z$  represents the number of small isolated superpixels to be merged.<sup>c</sup> Partially. <sup>d</sup>  $t$  is the number of relocations. <sup>e</sup> Without the saliency map computation. <sup>f</sup>  $d$  is the number of hierarchy levels. <sup>g</sup> Time complexity in HERs module. <sup>h</sup> Without parallelization.

### 3 BENCHMARK

#### 3.1 Superpixel methods

In this work, we identified 17 open source codes from the recent superpixel literature: **AINET** [126], **SIN** [138], **SSFCN** [134], **DISF** [14], **RSS** [21], **ODISF** [15], **IBIS** [17], **DRW** [57], **DAL-HERS** [89], **ISF** [116], **GMMSP** [6], **SCALP** [41], **SNIC** [2], **SH** [127], **LNSNet** [147], **SICLE** [11], and **LSC** [23, 67]. In addition, we include the 6 methods recommended as state-of-the-art in [109]: **SLIC** [1], **SEEDS** [114], **ERS** [71], **ETPS** [135], **CRS** [28], and **ERGC** [20]. Finally, a grid segmentation (**GRID**) was used as a baseline. Regarding implementation, we used the **SEEDS**, **CRS**, and **ERGC** code available in the benchmark of Stutz et al [109]. Also, we implemented grid segmentation. For the other methods, we use the original authors' code. Concerning the method's parameters, we use those recommended by the original works, since fine-tuning them may result in a worse parameter setting than the original ones, and tuning them for each dataset does not assess the methods' generalization ability. All evaluated methods allow some control over the number of superpixels generated. In our experiments, we assess segmentations with  $K \approx \{25, 50, 75, 100, 200, 300, 400, 500, 600, 700, 800, 900, 1000\}$  desired superpixels, except for the robustness evaluation, with only  $K \approx 400$  superpixels. In the following, we briefly present the superpixel methods used in our benchmark.

**SLIC** [1], **SCALP** [41], and **LSC** [67] perform *neighborhood-based* clustering, in which the clustering strategy comprises a distance function limited to a region concerning a reference point in the image. In these three methods, the reference point consists of the center of each cluster, and the search region size depends on the expected superpixel size. **SLIC** is an iterative method based on k-means whose superpixel centers start with a simple grid sampling and its distance measurement, which is based only on color, spatial position, and superpixel area, gives better control over the size and compactness of the superpixels. In **SCALP**, the distance function from a center to a pixel is weighted according to the linear path between these two points using a boundary map. On the other hand, **LSC** explores features at the pixel level, mapping them into 10-dimensional points. Similarly, **SNIC** [2] and **DRW** [57] use a *dynamic-center-update* clustering strategy. Based on **SLIC**, **SNIC** guarantees the connectivity of its superpixels during clustering and does not require multiple iterations. Conversely, **DRW** formulates the clustering problem based on the Random Walk algorithm [45] and adds dynamic nodes to the graph to reduce redundant computation and capture features at the region level.

**CRS** [28], **SEEDS** [114], **ETPS** [135], and **IBIS** [17] perform a *boundary evolution* clustering, which begins with a grid segmentation and updates the superpixel contours according to an energy function. **CRS** updates the pixel-superpixel assignment based on the image content and the *Gibbs-Markov random field* model, improving the segmentation through the iterations. In contrast, **SEEDS**, **ETPS**, and **IBIS** evaluate the superpixel boundaries with a coarse-to-fine strategy, explicitly dividing the image blocks. **SEEDS** uses an approach based on the *Hill-Climbing* algorithm and an optimization function with characteristics based on the color histogram. On the other hand, **ETPS** orders the superpixel boundaries evaluation using a priority queue. Its optimization function uses features at the pixel level to optimize homogeneity, compactness, size, and smoothness. Conversely, **IBIS** focuses on efficiency by employing a parallel coarse-to-fine strategy to optimize a **SLIC**-based distance function.

In contrast, the *path-based* clustering methods **ERGC** [20], **RSS** [21], **ISF** [116], **DISF** [14], **ODISF** [15], and **SICLE** [13] usually focus on boundary adherence instead of compactness. While **RSS** provides a non-iterative method that guarantees the optimality of the generated forest, **ISF**, **DISF**, and **ODISF** use iterative strategies. The **ISF** recalculates the position of the seeds at the end of each iteration. At the same time, **DISF**, **ODISF**, and **SICLE** perform an initial oversampling and

further remove the less relevant seeds at the end of each iteration. While **DISF** only uses pixel and path-based characteristics, **ODISF** and **SICLE** include saliency information in their removal step, but only **SICLE** allows to control the saliency importance. Unlike the others, **ERGC** formulates the segmentation with the *Eikonal* equation, solving it with the *Fast Marching Algorithm* [101], which calculates the minimum geodesic paths of the graph. Similar to the path-based methods, the *hierarchical* ones usually prioritize boundary adherence. However, most of them do not require several iterations to generate superpixels, and their hierarchical structure provides several segmentation levels (also called scales) with a unique execution. The **SH** [127] and **DAL-HERS** [89] produce a superpixel *hierarchy* according to locality and causality criteria [50]. While **SH** relies on *Boruvka's algorithm* [128], **DAL-HERS** generates affinity maps with a residual convolutional network and uses these maps to create a superpixel hierarchy.

Concerning methods that generate superpixels using *deep neural architecture*, **SSFCN** [134] and **AINET** [126], they employ u-shaped networks to extract soft pixel-superpixel assignment using a supervised strategy. The former directly outputs a soft pixel-superpixel association map, while the latter employs a new *Association Implantation* module to compute the soft assignment. **AINET** also uses a boundary-perceiving loss to improve boundary delineation. Conversely, **LNSNet** [147] relies on a non-iterative clustering module and employs a lifelong learning strategy that does not require ground truth labels. Unlike previous deep-based architectures, **SIN** [138] uses an interpolation network composed of fully convolutional layers to extract multi-layer features used in interpolation layers as association scores. These scores are used in multiple vertical and horizontal interpolation steps to expand the pixel-superpixel association matrix while enforcing spatial connectivity.

Finally, **GMMSP** [6] models the segmentation task as a weighted sum of Gaussians, each associated with a superpixel. On the other hand, **ERS** [71] models it using *Random Walk* [45] and generates superpixels from the cut in the image graph that optimizes its function.

### 3.2 Datasets

We selected five datasets which impose different challenges for superpixel segmentation: *Birds* [80]; *Insects* [80]; *Sky* [3]; *ECSSD* [106]; and *NYUV2* [107]. Table 3 summarizes their main characteristics. Birds, Insects, and Sky have natural images. The former contains images of birds, which have thin and elongated parts, hard to delineate with compact superpixels. When there are more birds, they often overlap, making it difficult to delineate them separately. In the Birds dataset, the background does not have a specific pattern and may (or may not) be blurred, colored, and textured. Similarly, the Insects dataset has images containing one or more insects with thin and elongated parts. Compared to the Birds dataset, it has more blurred and less textured backgrounds, and their objects (the insects) have thinner parts. Therefore, it has more challenging objects but less difficult backgrounds than the Birds dataset. In contrast, the Sky dataset has images with one plane each. Most images in the Sky dataset have large regions with low color and subtle luminosity variations. The ground

Table 3. Characteristics of the five datasets used in this work to evaluate superpixels.

	Birds	Insects	Sky	ECSSD	NYUV2
Image content	Natural	Natural	Natural	Natural and urban	Indoor
Number of images	150	130	60	1,000	1,449
Minimum image size	300 × 300	640 × 359	599 × 399	400 × 139	608 × 448
Maximum image size	640 × 640	640 × 640	825 × 600	400 × 400	608 × 448

truth in Birds, Insects, and Sky datasets are binary masks with only one connected object per image. The objects in Birds, Insects, and Sky, are a bird, an insect, and the sky, respectively.

In contrast with the aforementioned datasets, ECSSD and NYUV2 datasets have urban scenes. Specifically, the ECSSD dataset has images in both natural and urban environments, while NYUV2 is composed of video sequences from several indoor scenes recorded by Microsoft Kinect. The images in the ECSSD dataset have complex scenes, most with non-uniform regions and backgrounds composed of several parts. In the ECSSD dataset, the images may have more objects, many without well-defined boundaries. Furthermore, some objects have transparency, which makes them difficult to identify. Conversely, the RGBD images in the NYUV2 dataset have rich geometric structures with large planar surfaces, such as the floor, walls, and table tops. Its images also have small objects and occlusion, accentuated by the mess and disorder common in inhabited environments. In the ECSSD dataset the ground truth images are binary masks, each one with at least one connected object. In the NYUV2 dataset, the ground truth images have dense multi-class labels. However, for those in the NYUV2 dataset, we remove unlabeled pixels similar to [109].

### 3.3 Evaluation criteria

**3.3.1 Connectivity.** Connectivity is one of the most fundamental properties in superpixel segmentation. Superpixels are connected when their pixels form a connected component considering the X and Y axes. Also, some superpixel methods may consider pixels in diagonal on the X and Y axes as connected. However, several superpixel approaches fail to meet this property. Specifically, most methods with deep networks in their Main Processing step still struggle to produce superpixels, since they cannot provide a hard pixel-superpixel assignment. This work evaluates connectivity considering the eight neighbors on the X and Y axes – *i.e.*, including diagonals. We also perform a simple post-processing to ensure connectivity that gives a unique label to each connected component. Then, to maintain the generated number of superpixel labels, we merge the superpixel with fewer pixels and its most similar adjacent superpixel, considering the mean color. The merging step continues until the number of superpixels achieves the number of labels. For example, considering a method that generates 105 superpixel labels but has 200 connected components, the aforementioned post-processing step ensures 105 connected superpixels.

**3.3.2 Control over the number of superpixels.** Controlling the number of superpixels is also of utmost importance. In image segmentation, one may perform different segmentation to achieve distinct goals. For instance, object segmentation aims to divide images into object and background regions, while semantic segmentation densely labels pixels to each label associated with a semantic meaning. Conversely, in superpixel segmentation, each labeled region must have pixels with similar characteristics (usually color). Furthermore, one must delineate image objects by merging superpixels, and the number of superpixels may vary, typically being parameterized. Such control is important when using superpixel segmentation as preprocessing in other tasks. For instance, a very high number of superpixels may not effectively reduce redundant information or image primitives. Likewise, very few superpixels may result in lost important information due to non-homogeneous regions. In this work, we evaluate the number of superpixels generated considering the number of distinct labels.

**3.3.3 Boundary delineation.** Boundary adherence concerns the ability to superpixel borders to adhere to the object borders. Most superpixel methods evaluate their boundary adherence with quantitative and qualitative evaluation. In superpixel segmentation, the quantitative evaluation involves using ground truth images with binary masks or dense multi-class labels. However, such images may consider only some of the objects. Therefore, a quantitative analysis may be insufficient. Several measures evaluate superpixel boundary adherence, but some of them have

important drawbacks, while others have a high correlation [109]. Following Stutz et al. [109], we use *Boundary Recall* [81] and *Undersegmentation Error* [87] to evaluate boundary adherence.

**3.3.4 Color homogeneity.** Color homogeneity relates to the inner color similarity in superpixels. As the union of superpixels must be able to delineate image objects, more homogeneous superpixels tend to contain information from a single object. In superpixel literature, this property is quantitatively evaluated and is independent of the image ground truth. One may initially define color homogeneity as the simple color variation in superpixels concerning the image color variation. However, regions with textures such as grass to water are visually homogeneous, increasing the difficulty in homogeneity assessment. In this work, we quantitatively assess color homogeneity with *Explained Variation* [86] and *Similarity between Image and Reconstruction from Superpixels* [8].

**3.3.5 Compactness.** Compact superpixels have convex and regular shapes. Previous works demonstrate that highly adherent superpixels do not necessarily produce better segmentations [1, 87, 108, 109]. Specifically, Schick et al. [98] demonstrated an inverse and non-linear association between compactness and boundary adherence, and they argue that highly adherent superpixels are similar to overfitting the image boundaries, not necessarily capturing the most important ones. Compactness is usually quantitative and qualitatively assessed. In this work, we evaluate compactness using the *Compactness index* [98] and include its qualitative analysis in our visual quality criteria.

**3.3.6 Stability.** Stability is not a common evaluation criterion in superpixel segmentation, being first conducted in [87] to evaluate the superpixel stability of affine transformations. In [109], stability is redefined to consider stable the segmentation whose performance monotonically increases with the number of superpixels. In this sense, the minimum and maximum performances are considered the lower and upper bounds, while the standard deviation gives how much the overall performance varies. Following [109], we assess superpixel stability considering the minimum, maximum, and standard deviation of boundary adherence and color homogeneity measures.

**3.3.7 Robustness.** Similar to stability, evaluating robustness is uncommon in superpixel segmentation, being mostly performed in benchmark papers. In [87], robustness and stability are considered the same, and they refer to evaluating how much superpixels change when applying affine transformations. In [109], the concept of robustness was extended to the ability to maintain performance while increasing image blur and noise. Following [109], we evaluate robustness against noise by considering salt and pepper and average blur.

**3.3.8 Runtime.** Superpixels are widely used as pre-processing to reduce the workload and extract higher-level features. However, such benefits may be diminished with time-consuming superpixel algorithms. This problem is especially significant in tasks that require real-time execution. Therefore, the execution time is a crucial factor to consider in superpixel segmentation papers. In this work, we assess the execution time of CPU and GPU-based methods.

**3.3.9 Visual quality.** Superpixel papers often qualitatively assess to determine whether the quantitative results reflect segmentation quality. When evaluating superpixels, qualitative assessment is crucial as it captures aspects not covered by quantitative analysis due to the absence of a specific ground truth. Our visual quality assessment focuses on four key aspects: boundary adherence, compactness, smoothness, and regularity. Boundary adherence refers to the superpixels' ability to accurately delineate important image boundaries, regardless of the ground truth image. Conversely, compact superpixels have a regular and convex shape, whereas smoothness (or smooth boundaries) relates to the boundary length of superpixels. Moreover, regularity refers to their shape, size, and arrangement. A regular superpixel segmentation contains compact superpixels of similar size and approximately the same number of adjacent superpixels in both the X and Y image axes.

## 4 RESULTS

In this work, we evaluate 23 superpixel methods and a grid segmentation baseline according to different aspects. In Section 4.1, we quantitatively evaluate object delineation, color homogeneity, and compactness, summarizing these results with a distribution analysis using boxplots. Then, we evaluate the runtime in CPU and GPU in Section 4.2. In Section 4.3, we perform a qualitative evaluation concerning superpixel contour smoothness, compactness, and adherence to the object borders. The reader should refer to Appendix C for more experiments. In Appendix C.1, we analyze the number of generated superpixels and their connectivity. Appendix C.2 presents the stability assessment using the minimum (min), maximum (max), and standard deviation (std) of the measures BR, UE, EV, and SIRS. In addition, Appendix C.3 presents the robustness assessment against salt and pepper noise and average blur. According to the connectivity analysis in Appendix C.1, we include a merging step as post-processing on methods that do not guarantee connectivity to perform the experiments in Section 4.1 and Appendix C.2. Finally, in Appendix C.4, we discuss the overall performance of superpixel methods concerning their clustering category. In quantitative (Section 4.1), connectivity (Appendix C.1), and stability (Appendix C.2) results, we report the experiments on Birds, Insects, Sky, and ECSSD on the same plot since their results are similar. The superpixel methods and evaluation codes used in this work are available in our benchmark at <https://github.com/IMScience-PPGINF-PucMinas/superpixel-benchmark>.

### 4.1 Quantitative evaluation

**4.1.1 Object delineation.** As shown in Figure 5, **GRID**, **CRS**, and **SEEDS** reach the worst results in most datasets. According to the evaluation with UE, most methods have low leakage in Birds+Insects+Sky+ECSSD datasets (Figure 5), while the indoor images in NYUV2 are more challenging. Similarly, the delineation measured by BR is generally high, except in NYUV2. In Birds+Insects+Sky+ECSSD datasets, the best scores in both UE and BR were achieved by **SICLE**, **ODISF**, **DISF**, **LSC**, **ERS**, **GMMSP**, **ISF**, and **SH**. Furthermore, **SH**, **ISF**, and **RSS** achieved similar BR, but **RSS** has worse UE. The distribution of these results can also be observed in the boxplots (at the bottom of Figure 5).

In Birds+Insects+Sky+ECSSD datasets, **SICLE** and **ODISF** have the best BR and UE, followed by **DISF**, while **SH** and **RSS** have the best BR in NYUV2. Also, **ETPS** has the best UE in NYUV2. In contrast, **SICLE** and **ODISF** have poor performance in Sky and NYUV2 datasets, while the same occurs for **ETPS** in Birds+Insects+Sky+ECSSD datasets. In the Sky dataset, the poor delineation of **SICLE** and **ODISF** corresponds to a poor saliency map guiding the segmentation, whose importance can only be reduced in **SICLE**. In contrast, their poor performance in the NYUV2 dataset is due to these methods producing more superpixels in the image region identified as salient. This strategy is interesting when the salient region corresponds to the object of interest or when this region has more complex information, requiring more superpixels to obtain a better delineation. However, the ground truth in the NYUV2 dataset has multi-class labeling. As one may note in Figure 5, the leakage in the NYUV2 dataset is too high compared to the other datasets, resulting in similar UE results for most methods.

In most datasets, **RSS** and **SH** have competitive and similar BR results but worse UE. This observation is more evident in the boxplots (at the bottom of Figure 5). In contrast, **DAL-HERS**, **ETPS**, **IBIS**, and **SLIC** obtained a low delineation, only superior to **GRID**, **SEEDS**, and **CRS**. Their results are followed by **SNIC**, **SCALP**, **DRW**, and **LNSNet**. Also, according to the boxplot results (at the bottom of Figure 5), **LNSNet** seems to have the best UE in the NYUV2 dataset, but it achieves low leakage when the number of superpixels is too high (see the control over the number of superpixels in Appendix C.1).

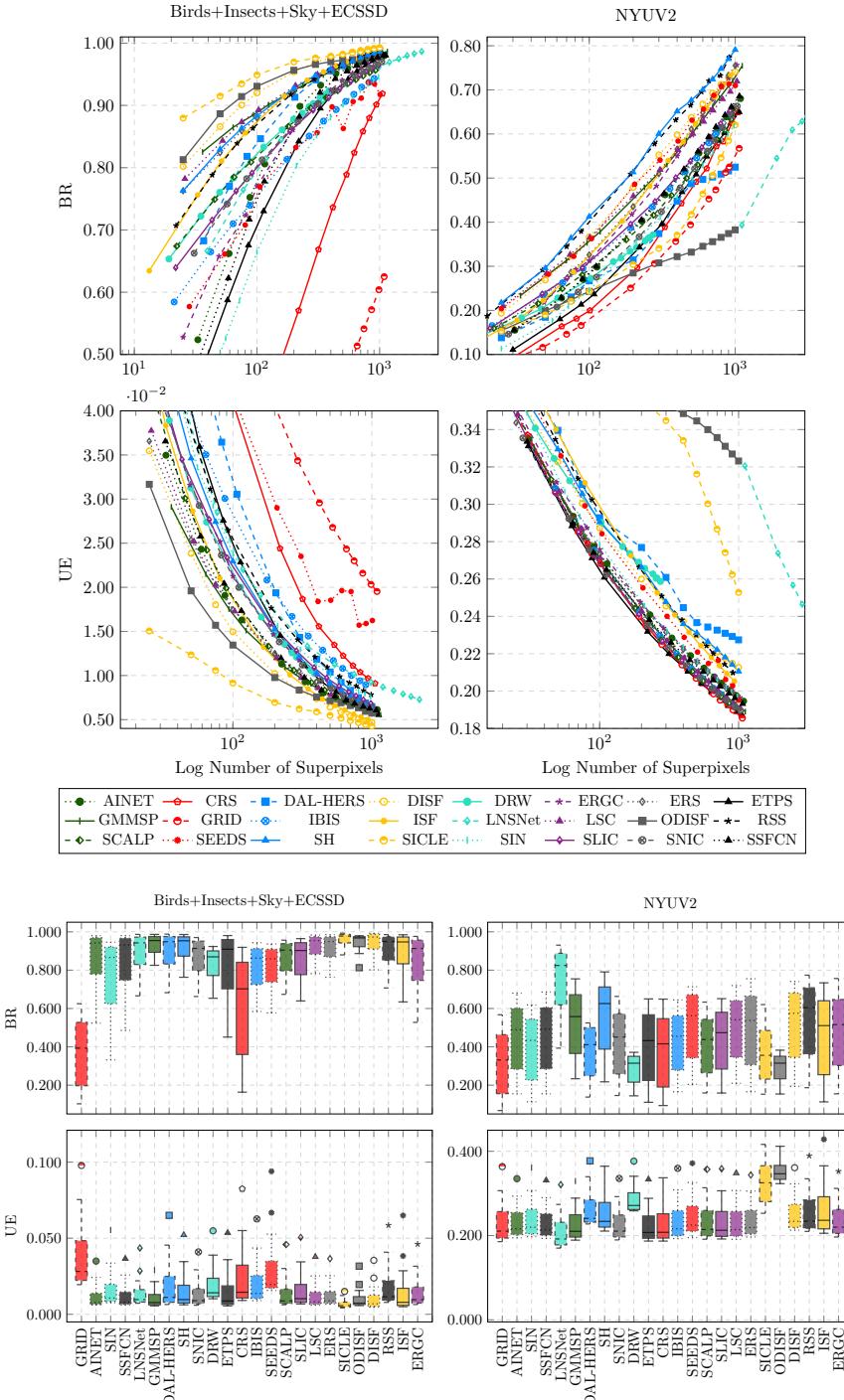


Fig. 5. Results for BR and UE on Birds+Insects+Sky+ECSSD and NYUV2 datasets.

**4.1.2 Color homogeneity.** When evaluating the color homogeneity (Figure 6) with EV and SIRS, the results of the first measure are generally higher and closer to each other compared to the second one. However, their results show some similarities. **GRID** and **CRS** have the worst results in all datasets in both measures, followed by **ODISF** and **SICLE**. Among these methods, only **ODISF** and **SICLE** have an accurate delineation, and their low color homogeneity results from fewer superpixels in the non-salient image regions. Conversely, **DISF** has the best results in most datasets, followed by **SH** and **LSC**. Although they have different clustering approaches, all these methods have high boundary adherence. **ISF**, **RSS**, **SCALP**, and **GMMSP** also achieve competitive color homogeneity results. The distribution of these results can also be observed in the boxplots (at the bottom of Figure 6).

**4.1.3 Compactness.** Figure 7 shows the compactness evaluation. As expected, **GRID** obtains the most compact segmentations. Aside from **GRID**, **CRS**, **SIN**, and **ETPS** have the highest compactness across the datasets, followed by **SCALP**, **SNIC**, **AINET**, and **SSFCN**. Also, **SLIC** and **IBIS** achieve similar compactness, usually lower than **AINET** and **SSFCN**. The label propagating strategy in **AINET**, **SSFCN**, and **SIN** favors compactness. In contrast, **CRS**, **SCALP**, **SNIC**, and **SLIC** have a parameter to control compactness, while the initial grid segmentation in **ETPS** and **IBIS** provides initial compact superpixels. While **CRS** and **ETPS** produce superpixels by optimizing the contours of a grid segmentation, the others use different approaches based on **SLIC**. In contrast, **LSC** and **GMMSP** present similar and moderate compactness. Among the evaluated methods, only **SEEDS** have high variability in compactness. More delineation-focused methods, such as **SICLE**, **ODISF**, **DISF**, **SH**, and **DAL-HERS**, produced less compact segmentations.

**4.1.4 Overall.** As one may see in Figures 5, 6, and 7, most methods with *path-based* clustering (**ERGC**, **ISF**, **DISF**, **RSS**, **ODISF**, and **SICLE**) have similar performance in object delineation, compactness, and homogeneity. They usually achieve high delineation but low compactness. Similarly, the method with *graph-based* clustering (**ERS**) performs clustering based on graphs, having a competitive (but not the best) delineation on all datasets with more compactness than *path-based* methods. On the other hand, *neighborhood-based* clustering approaches (**SLIC**, **LSC**, and **SCALP**) have more varied results. While **LSC** achieves excellent delineation and more homogeneous superpixels, **SLIC** has moderate compactness and worse delineation. Conversely, **SCALP** has better delineation, color homogeneity, and compactness than **SLIC** but lower delineation and color homogeneity than **LSC**. Methods with *boundary evolution* clustering (**SEEDS**, **IBIS**, **CRS**, and **ETPS**) perform clustering based on contour optimization, achieving different results due to the distinction between their optimization functions. These methods produce the worst results in object delineation and color homogeneity but with higher compactness. Among them, **IBIS** achieves similar object delineation and color homogeneity to **SLIC**, whereas **CRS** and **SEEDS** have the worst delineation and homogeneity but greater compactness among all methods.

Methods with a *dynamic-center-update* clustering (**DRW** and **SNIC**) use strategies to adapt the number of generated superpixels to the image content. Despite their similarities, **DRW** and **SNIC** use different features and optimization functions, which explains the contrast in their results. While **DRW** has better delineation and fewer superpixels, **SNIC** generates more compact and homogeneous superpixels. Also, the lower color homogeneity of **DRW** compared to **SNIC** is due to the smaller number of superpixels produced by **DRW**. Concerning *hierarchical* approaches (**SH** and **DAL-HERS**), they have low compacity and high color homogeneity. However, **SH** has competitive delineation and color homogeneity, while **DAL-HERS** has worse results. On the other hand, the method with *data distribution-based* clustering (**GMMSP**) has a performance similar to **LSC**. **GMMSP** have moderate compactness, producing more compact superpixels than *path-based* clustering methods but less than most *neighborhood-based* ones. Finally, methods that perform

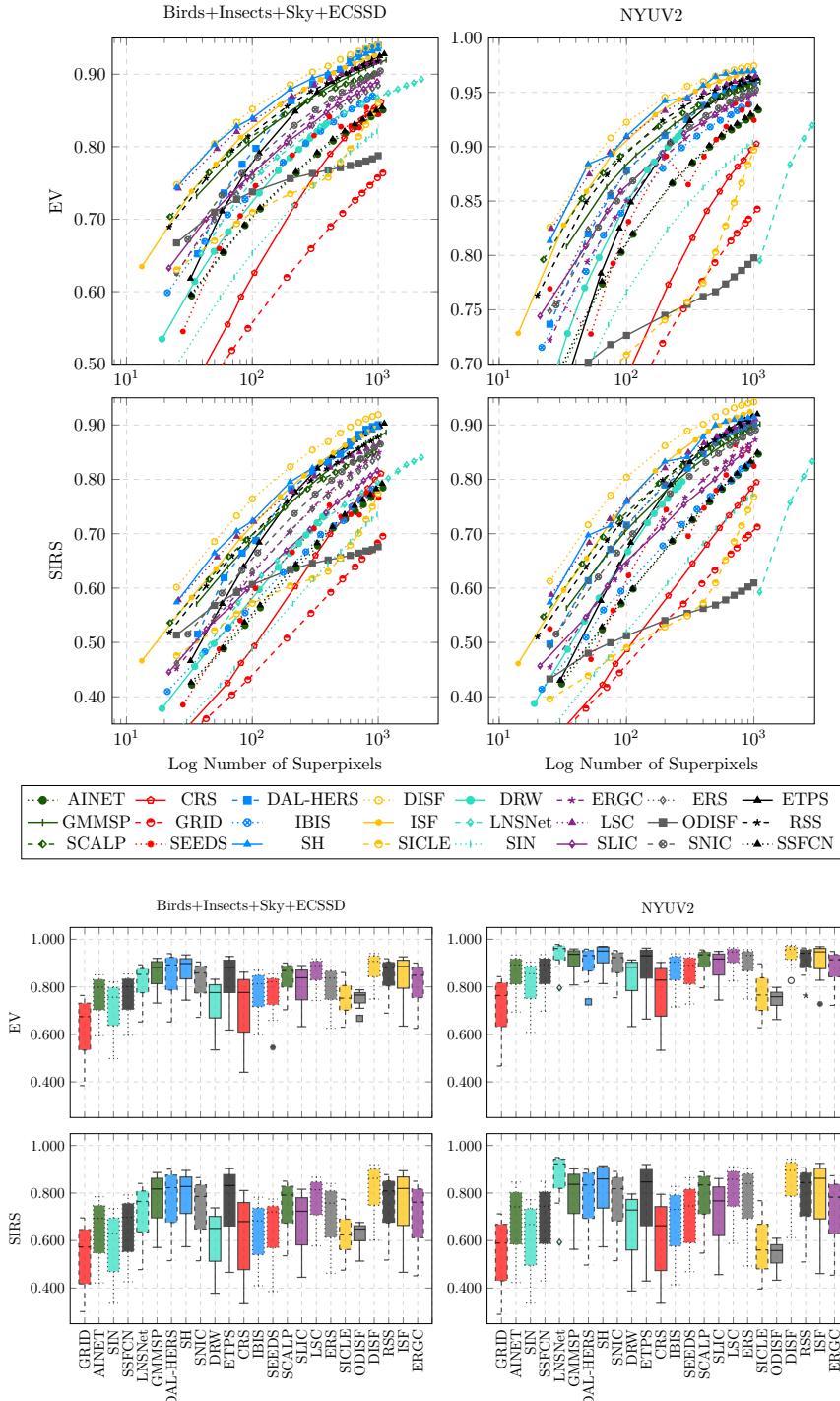


Fig. 6. Results for EV and SIRS on Birds+Insects+Sky+ECSSD and NYUV2 datasets.

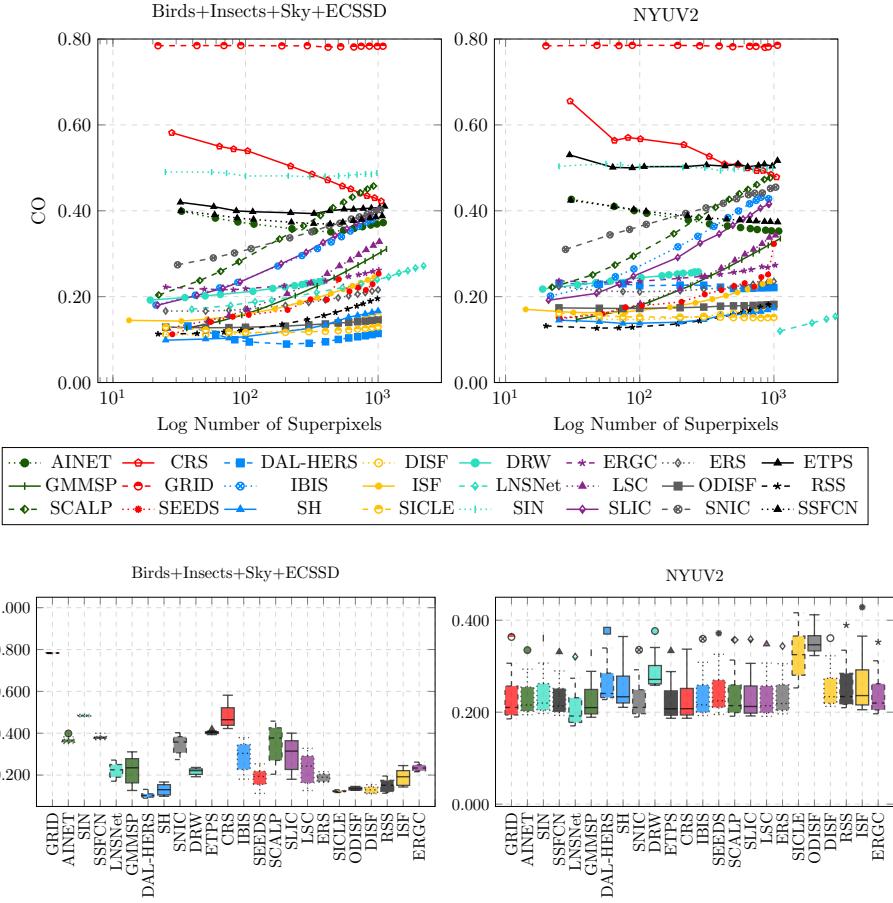


Fig. 7. Results for CO on Birds+Insects+Sky+ECSSD and NYUV2 datasets.

clustering with *deep architectures* have varied performances. Using u-shaped architectures, **SSFCN** and **AINET** have similar and moderate results in all metrics. On the other hand, **LNSNet** has excellent BR, but its UE indicates more leakage. Finally, although the *interpolation network* in **SIN** guarantees connected superpixels, they have poor delineation and color homogeneity but high compactness.

## 4.2 Runtime

Execution time may be critical in superpixel methods, especially for real-time applications. Figure 8 shows the CPU<sup>1</sup> and GPU<sup>2</sup> time in seconds without the post-processing of Section C.1. For **SCALP**, **ODISF**, and **SICLE**, we do not include the edge maps and saliency maps computation. As one may see in Figure 8, due to the images of the ECSSD dataset being generally smaller than the others, the runtime in this dataset is usually shorter. Therefore, we merged the results of the datasets Birds, Insects, and Sky since they were similar.

<sup>1</sup>CPU Intel® Core™ i5-7200U @ 2.5GHz x 4, 64bit with 24GB RAM.

<sup>2</sup>CPU Intel® Core™ i7-8700 @ 3.20GHz x 12, 64bit with 31GB RAM and a GPU Nvidia GeForce GTX 1080 with 8GB RAM.

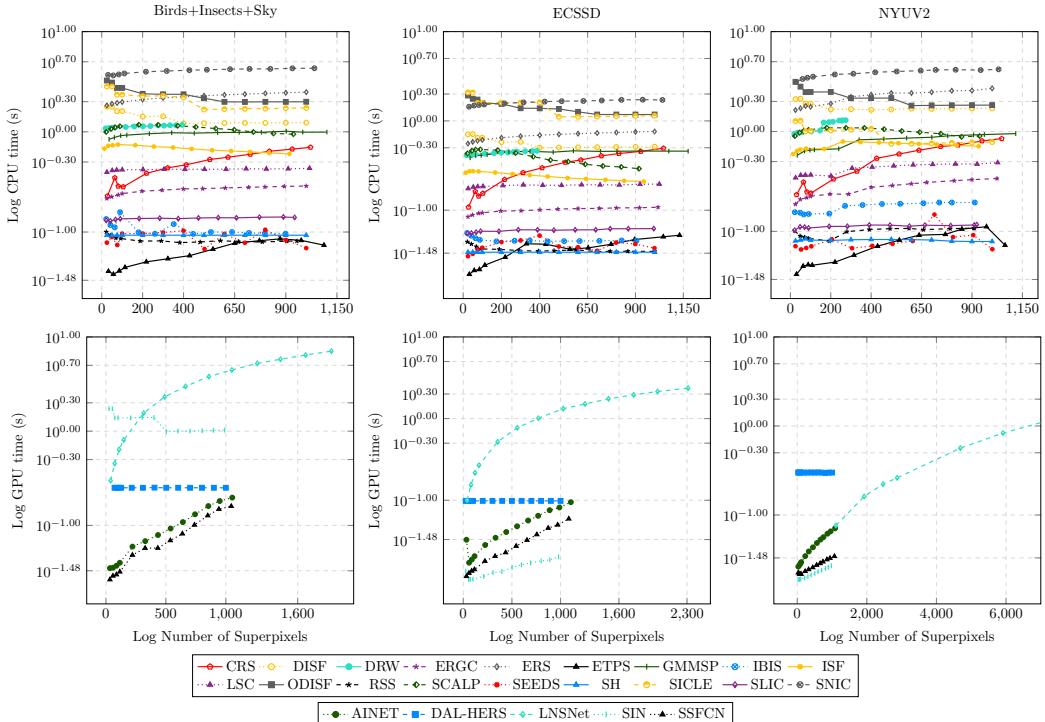


Fig. 8. Runtime in seconds on Birds+Insects+Sky, ECSSD, and NYUV2 datasets.

According to the CPU runtime (on the first row in Figure 8), methods with *boundary evolution* clustering (except **CRS**) achieve the lowest execution time (around 0.05 seconds), followed by **SH** (which performs *hierarchical* clustering) with around 0.09 seconds. **SLIC** has similar efficiency, requiring around 0.14 seconds per image, while the remaining *neighborhood-based* clustering methods (**LSC** and **SCALP**) vary in efficiency. **LSC** requires around 0.4 seconds per image on Birds+Insects+Sky and NYUV2 datasets, while **SCALP** needs approximately twice as long. *Path-based* clustering methods (**ERGC**, **ISF**, **RSS**, **DISF**, **SICLE**, and **ODISF**) also show varied efficiency. **SICLE**, **ODISF**, and **DISF** achieve higher runtimes, while **ISF** and **ERGC** require less than 1 second per image. In contrast, **RSS** has a competitive execution time (less than 0.1 seconds).

Methods with a *dynamic center update* clustering (**DRW** and **SNIC**) also have distinct runtimes. While **DRW** requires around 1 second per image, **SNIC** is the most time-consuming on a CPU. Considering *graph-based* clustering, **ERS** requires a high execution time, similar to **ODISF**. **GMMS**, the only one with clustering based on *data distribution*, achieves similar efficiency to **SCALP**. As one may see in Figure 8, **DAL-HERS**, **SSFNC**, **AINET**, **LNSNet**, and **SIN** were executed on a GPU. **LNSNet** has the worst execution time of all evaluated methods, and it increases according to the number of superpixels. **SSFNC**, **AINET**, and **SIN** have similar behavior, but their running times are much lower. Among these, **SIN** is usually faster, except in the Birds dataset. One may argue that the Birds dataset has more stretched images than the other datasets, which may hinder the interpolation operations. In contrast, **DAL-HERS** has an excellent execution time, requiring less than 0.3 seconds per image. **SH** and **DAL-HERS** are the only ones whose execution time is constant

since they produce a *hierarchy* of superpixels in a single execution. From cuts on the hierarchy, they produce different numbers of superpixels.

### 4.3 Qualitative evaluation

In this section, we evaluate the segmentations' visual quality regardless of their ground truth since the image object may vary according to the application. We assess visual quality based on the superpixels' adherence to the image boundaries, smoothness, compactness, and regularity. The smoothness is inversely related to the superpixel's boundary length. On the other hand, the superpixels' compactness relates to their area. Moreover, regularity refers to their shape, size, and arrangement. Figures 9, 10, and 11 present segmentations with approximately 100 and 700 superpixels on Birds, Insects, Sky, ECSSD, and NYUV2 datasets.

**4.3.1 Path-based clustering.** Relative to path-based clustering methods, **DISF**, **ODISF**, and **SICLE** produce superpixels with no compactness but competitive delineation. **ODISF** and **SICLE** produce more superpixels on the salient area identified by the saliency map, which can improve the delineation of this region. Due to this, there is a low number of superpixels in regions with low saliency, leading to a worse delineation in these regions but a superior delineation in the salient ones. Also, by fine-tuning the saliency maps, their results can improve. Comparing **SICLE** and **ODISF**, one can observe more accurate delineation in **SICLE** segmentations. On the other hand, **RSS**, **ISF**, and **ERGC** have some compactness and smooth contours. However, **RSS** cannot produce compact superpixels in very homogeneous regions, and its superpixels have a high variance in size. In contrast, **ISF** produces regular superpixels in homogeneous regions, but it has a high sensitivity to color variations, leading to non-smooth superpixels, highly variable in size, in more complex regions. Similarly, **ERGC** has good adherence to the object boundaries, and its superpixels have some regularity, without significant variations in size.

**4.3.2 Neighborhood-based clustering.** Regarding the neighborhood-based methods, **SLIC** produces very compact superpixels with good adherence to the image boundaries and more regularity in homogeneous regions. Although both delineation and compactness reduce for a lower number of superpixels, the delineation of **SLIC** is more compromised. In contrast, **SCALP** produces superpixels with excellent delineation that are more compact, smooth, and regular than **SLIC**. Although it produces less compact superpixels when the number of them reduces, the superpixels' contours of **SCALP** remain smooth. Unlike **SLIC** and **SCALP**, **LSC** only produces smooth superpixels in more homogeneous regions. However, it is sensitive to minor color variations, reducing its smoothness and compactness in regions with simpler textures. Furthermore, **LSC** may generate more elongated and thin superpixels at the strong image boundaries, obtaining an excellent delineation but with no compactness.

**4.3.3 Dynamic center update clustering.** With visually very similar segmentation to **SLIC**, **SNIC** also produces superpixels with high compactness and better delineation. In contrast, **DRW** does not generate compact superpixels and produces noticeably fewer superpixels than expected, especially in homogeneous regions. Despite this, it generates superpixels with good adherence.

**4.3.4 Boundary evolution clustering.** Similarly to **DRW**, **SEEDS** (Figure 10) creates superpixels with poor compactness and non-smooth boundaries. They have moderate delineation with small leakage regions, which significantly reduces when the number of superpixels reduces. In contrast to **SEEDS**, **CRS** generates highly compact and regular superpixels but with low boundary adherence. Similarly, **ETPS** produces very regular, smooth, and compact superpixels. However, its compactness, smoothness, and regularity slightly reduce for lower superpixels, while the delineation suffers

drastically. **IBIS** also generates significantly compact superpixels, whose compactness and smoothness vary depending on the region's homogeneity. Also, it produces regular superpixels at the homogeneous image regions. However, its sensitivity to color variations reduces compactness and smoothness in less homogeneous areas. Also, for lower superpixels, its adherence to contours is significantly reduced.

**4.3.5 Hierarchical clustering.** Regarding the hierarchical methods, **SH** has an excellent delineation, but its superpixels are not regular nor compact and have non-smooth contours in more complex regions. In addition, it generates elongated and thin superpixels at some of the strong image boundaries. **DAL-HERS** also has superb delineation but generates rough superpixels and some tiny ones, resulting in visibly poor segmentation.

**4.3.6 Deep-based clustering.** Methods that perform clustering using deep learning usually have moderate delineation and high compactness. **AINET** and **SSFCN** produce smooth superpixels with some regularity. Also, **AINET** produces less smooth superpixels than **SSFCN**, indicating more sensitivity to low color variation. In contrast, **LNSNet** generates a significantly higher number of superpixels than desired. Similarly to **ISF**, **LNSNet** produces compact superpixels in homogeneous regions, but its sensitivity to color variations implies very rough superpixels. It has good delineation when the number of superpixels is higher. However, their non-smooth contours do not have a high delineation even in regions with a more prominent boundary, causing small leaks. On the other hand, **SIN** has much more compact and regular superpixels than the other deep learning strategies, but its delineation often misses strong boundaries.

**4.3.7 Others.** The superpixels in **ERS** have good boundary adherence and do not vary much in size, but they have low smoothness. In comparison, **GMMSP** produces significantly more compact superpixels in homogeneous regions. In less homogeneous ones, **GMMSP** produces fewer compact superpixels but usually with smoother contours. By reducing the number of superpixels, the compactness of the less homogeneous image region barely changes. However, the compactness and smoothness drastically reduce in less homogeneous regions.

**4.3.8 Overall.** As one may see, **CRS**, **ERGC**, **ETPS**, **SCALP**, **SLIC**, **SNIC**, **IBIS**, **GMMSP**, **SSFCN**, **AINET**, and **SIN** produce visibly smooth, compact, and regular superpixels. These properties are most noticeable on **CRS**, **SCALP**, and **ETPS**. Nevertheless, high compactness may lead to a worse delineation, as in **CRS**, **SIN**, and **ETPS**. Conversely, **ERGC**, **IBIS**, **SNIC**, **SLIC**, **AINET**, and **SSFCN** had moderate boundary adherence, with worse results on smooth image boundaries. Among these methods, only **SCALP** and **GMMSP** achieved excellent boundary delineation. Concerning methods with less (or no) compactness and smoothness, **LNSNet** and **SEEDS** have the worst delineations. In contrast, **DRW**, **ERS**, **RSS**, **LSC**, and **ISF** have some compactness, smoothness, and good boundary adherence. One may observe the best delineation in **DISF**, **LSC**, **ODISF**, **SICLE**, **ISF**, and **SH**, although most do not have compactness or regularity. In particular, **DISF**, **ODISF**, **SICLE**, and **GMMSP** have visually better boundary adherence. Also, **ODISF** and **SICLE** use a saliency map to guide the segmentation, generating more superpixels in salient regions and fewer superpixels in non-salient ones, reducing the superpixel color homogeneity. As observed in the quantitative evaluation, when the saliency map corresponds to the desired object, the **ODISF**'s and **SICLE**'s delineations outperform the other methods, indicating that decreasing the saliency map importance may improve such results, which is only possible in **SICLE**.

Among the main processing categories, methods with contour evolution-based clustering usually produce the most compact and regular superpixels, although they have low boundary adherence. Conversely, those with neighborhood-based clustering usually have good delineation with high compactness and regularity. Similarly, dynamic-center-update clustering methods also achieve good

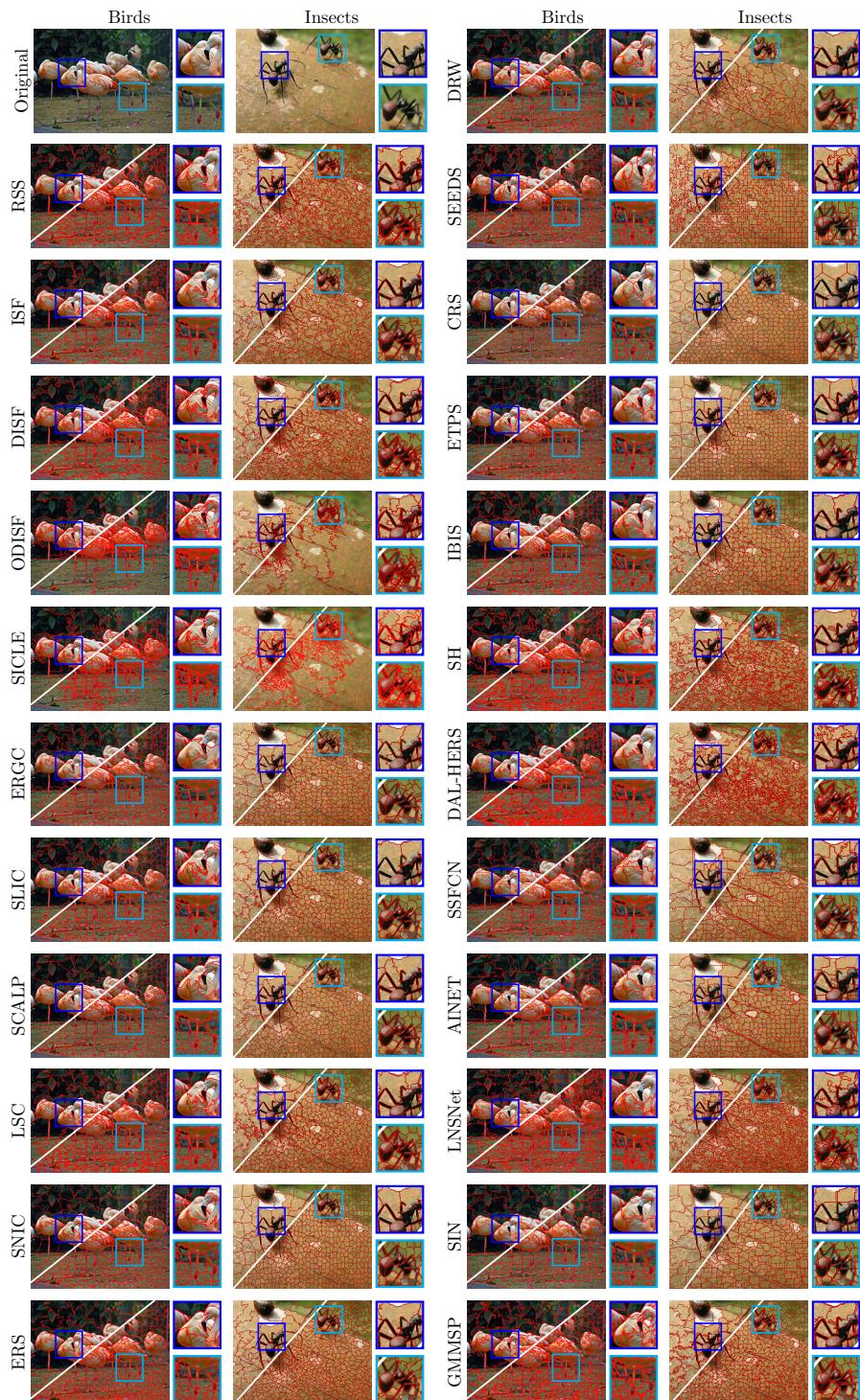


Fig. 9. Segmentation comparison with images from Birds, and Insects with 100 and 700 superpixels.

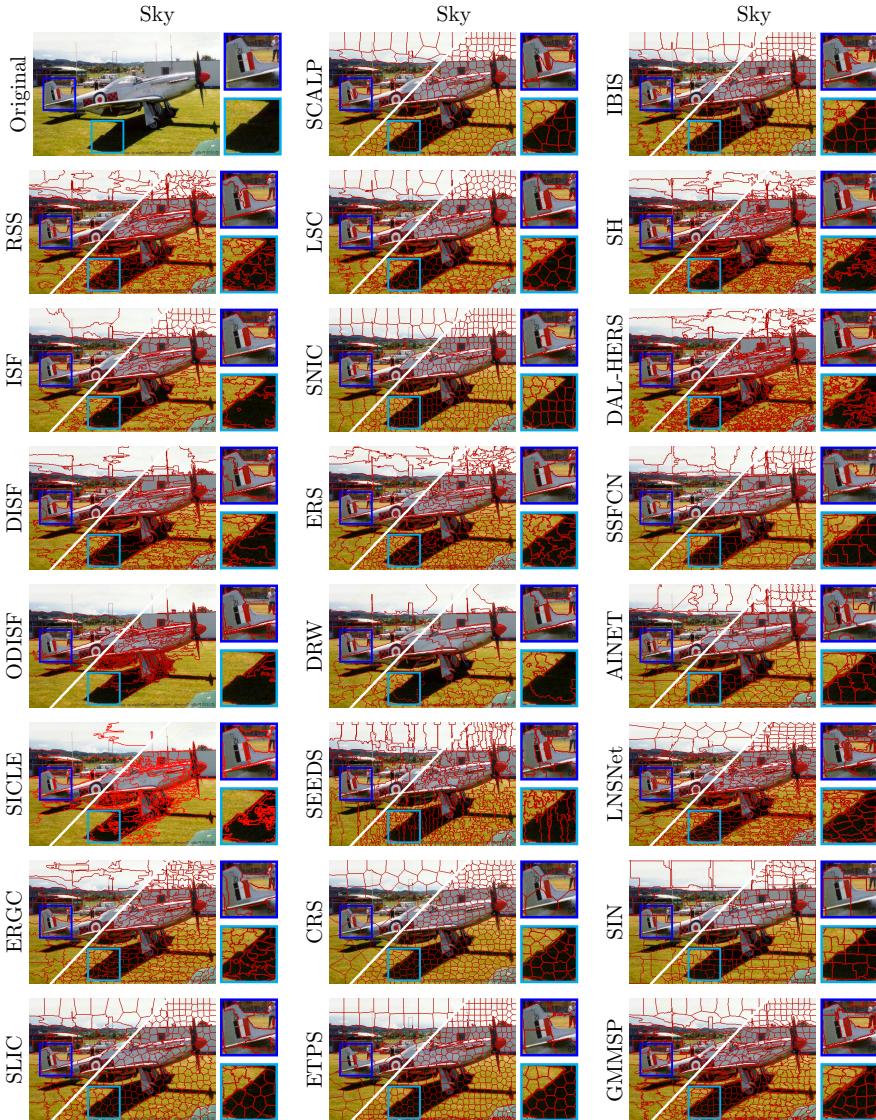


Fig. 10. Segmentation comparison with images from Sky with 100 and 700 superpixels.

boundary adherence. However, only **SNIC** shows high compactness and regularity, whereas **DRW** only has smooth superpixel contours. Finally, **GMMS** has great compactness and competitive delineation. Hierarchical methods, path-based methods, and **ERS** produce superpixels with low compactness. Also, the superpixels on hierarchical methods are neither compact nor smooth. Conversely, **ERS** and most path-based methods generate superpixels with low compactness and smoothness but with excellent boundary adherence. Methods with deep-based clustering have variate results in all criteria. Like neighborhood-based methods, **SSFCN** and **AINET**, both with u-shaped architectures but distinct loss functions and clustering layers, have moderate delineation and good compactness, with similar results across all criteria. In contrast, **SIN** produces much

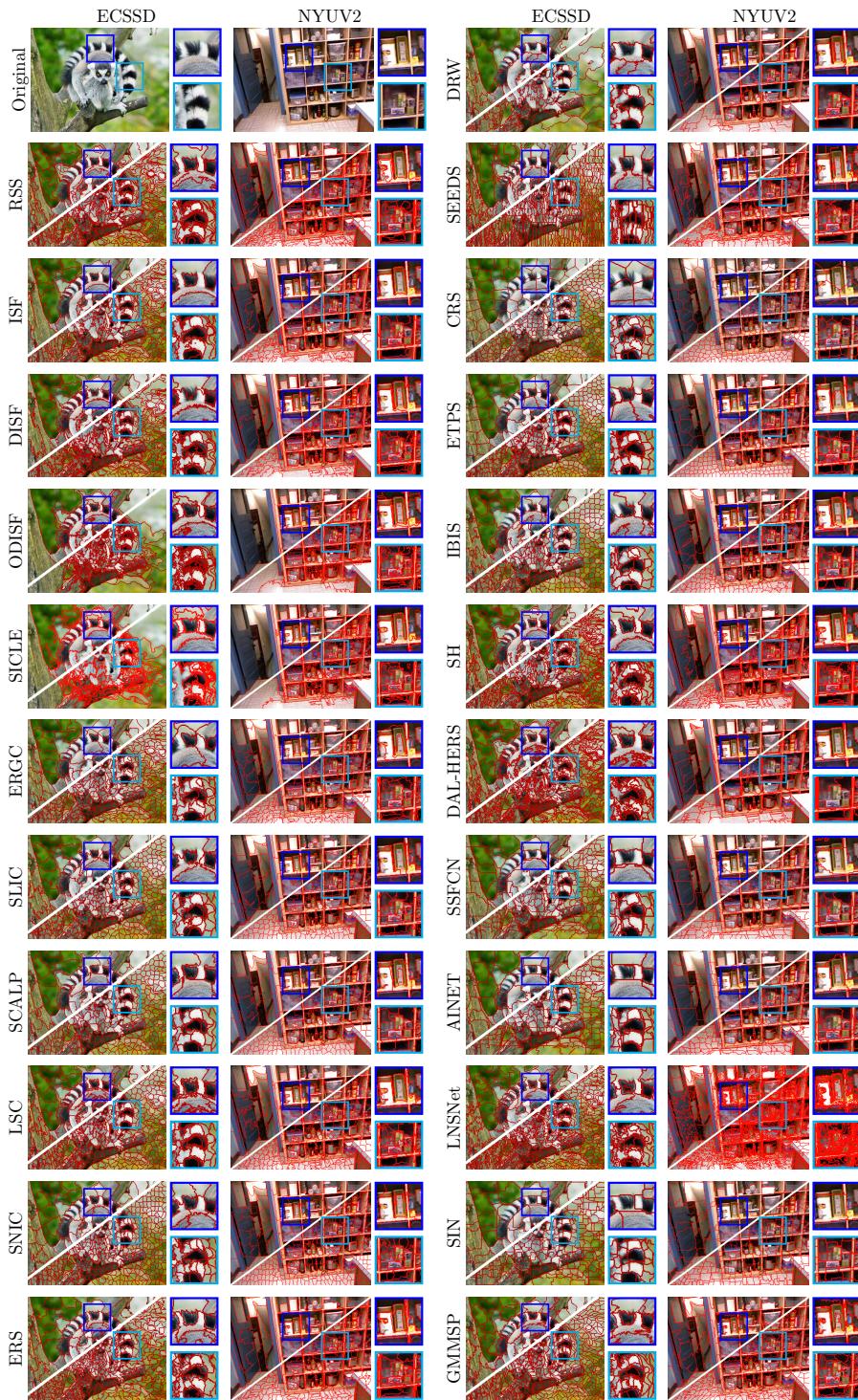


Fig. 11. Segmentation comparison with images from ECSSD and NYUV2 with 100 and 700 superpixels.

more compact and regular superpixels, but with low delineation. The regularity in **SIN** may be the result of its interpolation operations, which propagate superpixel labels to neighbor pixels considering the image lattice. On the other hand, **LNSNet** produces much more superpixels than desired, resulting in superpixels with no compactness or smoothness.

## 5 CONCLUSIONS

In this work, we present a taxonomy for superpixel methods, which categorizes them according to their processing steps and the level of abstraction of the features used. Our taxonomy separates each superpixel approach into up to three processing steps and categorizes the task performed at each one. Along with our taxonomy, we inform other important properties of 59 of the most recently and commonly used superpixel methods. We also provide a comprehensive literature review encompassing these methods. We present an extensive comparison among 23 superpixel methods and a grid baseline considering: superpixel connectivity, control of the number of superpixels, compactness, adherence to object contours, color homogeneity, stability, robustness to noise and blur, execution time, and visual quality.

According to our experiments, methods with clustering based on boundary evolution generally present greater efficiency, compactness, and regularity. Nevertheless, they have worse boundary adherence and color homogeneity. In contrast, methods with dynamic-update-clustering are less efficient and generate slightly less compact and regular superpixels. Also, they have better delineation and homogeneity than those based on boundary evolution. Conversely, methods with neighborhood-based clustering present more varied performances. For instance, **LSC** achieves good boundary adherence, compactness, and smoothness, while **SLIC** and **SCALP** have higher compactness but worse delineation. **GMMSP**, which performs clustering based on data distribution, obtains a competitive delineation, good compactness, and smooth superpixel contours, although no regularity. In addition, **ERS**, the only evaluated method with graph-based clustering, has a similar delineation but worse efficiency, compactness, and color homogeneity. Hierarchical methods also produce superpixels with excellent boundary adherence. In addition, they have low execution time, but their superpixels are neither visually compact nor regular. In contrast, deep learning-based methods achieve moderate or low delineation and varied compactness, but most are efficient, requiring around 0.1 seconds to process an image. In our evaluation, the path-based clustering methods generally have the best delineation and the most homogeneous superpixels. However, they have varied efficiency, low compactness, and low smoothness.

Most evaluated methods ensure superpixel connectivity and generate superpixels in a similar number to the desired one. In particular, **DISF**, **ODISF**, **SICLE**, **SH**, and **ERS** generated the exact number of desired superpixels. In contrast, only **LNSNet**, **SEEDS**, **CRS**, and **DRW** may produce unconnected superpixels. **LNSNet** creates almost twice the superpixels, many of these disconnected. On the other hand, the number of superpixels produced by **DRW** is usually lower than desired. Furthermore, in deep-based learning methods, the network usually cannot output connected superpixels but only soft pixel-superpixel assignments. These methods rely on post-processing to compute the hard assignment. Only **SIN** can directly output connected superpixels, but its label propagation mechanism cannot produce highly boundary-adherent superpixels. When evaluating robustness, most methods achieve good robustness to noise and blur. The worst results were observed in **DAL-HERS**, followed by **SICLE**, **LNSNet**, and **ERS**. In contrast, the most robust methods are **DISF**, **ERGC**, **RSS**, **ISF**, **ODISF**, **SEEDS**, and **SH**. We could also see that some methods produce a different number of superpixels according to the addition of noise or blur. For  $K \approx 400$ , **LNSNet** has more sensitivity in this criterion, creating more than 30,000 superpixels. **DAL-HERS** also produces significantly more superpixels, reaching almost 1,500 when increasing noise. In

addition, the number of superpixels produced by **IBIS**, **DRW**, **SLIC**, **GMMSP**, and **SCALP** produce quantities of superpixels close to the desired ones when increasing noise or blur.

Due to the trade-off between delineation and compactness, it is hard to establish which method has the best performance. Considering object delineation and color homogeneity, **DISF**, **ISF**, **LSC**, **GMMSP**, and **SH** have the best average performance and stability. **SH** has greater efficiency, followed by **LSC** and **ISF**. On the other hand, **GMMSP** has more compact superpixels, followed by **ISF**. When delineation and homogeneity are more critical than compactness, **DISF** is the most recommended. We also recommend **SICLE** and **ODISF** when only object delineation is crucial. Despite not having good results when the saliency map does not find the desired object, their superior performance in other datasets may indicate that fine-tuning the saliency detector can improve the results. However, for greater compactness at the expense of delineation, both **SCALP** and **SLIC** are recommended. Between these, **SLIC** has more compactness and low execution time but worse delineation and less color homogeneity. Considering real-time applications, only **SIN** could achieve around 30fps on GPU in most datasets. In smaller images (from ECSSD), **ETPS**, **SH**, and **RSS** have real-time results. Among these, **ETPS** produces more compact superpixels but with low delineation, while **SH** and **RSS** performed similarly in all criteria with high boundary adherence but low compactness.

## ACKNOWLEDGMENTS

The authors thank the Conselho Nacional de Desenvolvimento Científico e Tecnológico – CNPq – (Universal 407242/2021-0, PQ 303808/2018-7, 310075/2019-0), the Fundação de Amparo a Pesquisa do Estado de Minas Gerais – FAPEMIG – (PPM-00006-18), the Fundação de Amparo a Pesquisa do Estado de São Paulo – FAPESP – (2014/12236-1) and the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (COFECUB 88887.191730/2018-00 and Finance code 001) for the financial support.

## REFERENCES

- [1] Radhakrishna Achanta, Appu Shaji, Kevin Smith, Aurelien Lucchi, Pascal Fua, and Sabine Süsstrunk. 2012. SLIC Superpixels Compared to State-of-the-Art Superpixel Methods. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 34, 11 (2012), 2274–2282. <https://doi.org/10.1109/TPAMI.2012.120>
- [2] Radhakrishna Achanta and Sabine Süsstrunk. 2017. Superpixels and Polygons Using Simple Non-iterative Clustering. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 4895–4904. <https://doi.org/10.1109/CVPR.2017.520>
- [3] Eduardo Barreto Alexandre, Ananda Shankar Chowdhury, Alexandre Xavier Falcao, and Paulo A. Vechiatto Miranda. 2015. IFT-SLIC: A General Framework for Superpixel Generation Based on Simple Linear Iterative Clustering and Image Foresting Transform. In *2015 28th SIBGRAPI Conference on Graphics, Patterns and Images*. 337–344. <https://doi.org/10.1109/SIBGRAPI.2015.20>
- [4] Jianqiao An, Yucheng Shi, Yahong Han, Meijun Sun, and Qi Tian. 2020. Extract and Merge: Superpixel Segmentation with Regional Attributes. In *European Conference on Computer Vision*. Springer, 155–170.
- [5] F. Aurenhammer. 1987. Power Diagrams: Properties, Algorithms and Applications. *SIAM J. Comput.* 16, 1 (1987), 78–96. <https://doi.org/10.1137/0216006>
- [6] Zhihua Ban, Jianguo Liu, and Li Cao. 2018. Superpixel Segmentation Using Gaussian Mixture Model. *IEEE Transactions on Image Processing* 27, 8 (2018), 4105–4117. <https://doi.org/10.1109/TIP.2018.2836306>
- [7] Zhihua Ban, Jianguo Liu, and Jeremy Fouriaux. 2020. GMMSP on GPU. *Journal of Real-Time Image Processing* 17, 2 (2020), 245–257.
- [8] Isabela B Barcelos, Felipe De C Belém, Leonardo De M João, Alexandre X Falcão, and Guimarães Silvio JF. 2022. Improving color homogeneity measure in superpixel segmentation assessment. In *2022 35th SIBGRAPI Conference on Graphics, Patterns and Images (SIBGRAPI)*, Vol. 1. 79–84. <https://doi.org/10.1109/SIBGRAPI55357.2022.9991772>
- [9] Hans H.C. Bejar, Silvio Jamil Ferzoli Guimarães, and Paulo A.V. Miranda. 2020. Efficient hierarchical graph partitioning for image segmentation by optimum oriented cuts. *Pattern Recognition Letters* 131 (2020), 185–192. <https://doi.org/10.1016/j.patrec.2020.01.008>

- [10] Hans HC Bejar, Lucy AC Mansilla, and Paulo AV Miranda. 2018. Efficient unsupervised image segmentation by optimum cuts in graphs. In *Iberoamerican Congress on Pattern Recognition*. Springer, 359–367.
- [11] Felipe Belém, Isabela Borlido, Leonardo João, Benjamin Perret, Jean Cousty, Silvio JF Guimarães, and Alexandre Falcão. 2022. Fast and Effective Superpixel Segmentation Using Accurate Saliency Estimation. In *International Conference on Discrete Geometry and Mathematical Morphology*. Springer, 261–273.
- [12] Felipe Belém, Silvio Jamil F Guimarães, and Alexandre Xavier Falcão. 2018. Superpixel segmentation by object-based iterative spanning forest. In *Iberoamerican Congress on Pattern Recognition*. Springer, 334–341.
- [13] Felipe Belém, Benjamin Perret, Jean Cousty, Silvio JF Guimarães, and Alexandre Falcão. 2022. Efficient Multiscale Object-based Superpixel Framework. *arXiv preprint arXiv:2204.03533* (2022).
- [14] Felipe C. Belém, Silvio Jamil F. Guimarães, and Alexandre X. Falcão. 2020. Superpixel Segmentation Using Dynamic and Iterative Spanning Forest. *IEEE Signal Processing Letters* 27 (2020), 1440–1444. <https://doi.org/10.1109/LSP.2020.3015433>
- [15] Felipe C Belém, Benjamin Perret, Jean Cousty, Silvio JF Guimarães, and Alexandre X Falcão. 2021. Towards a Simple and Efficient Object-based Superpixel Delineation Framework. In *2021 34th SIBGRAPI Conference on Graphics, Patterns and Images (SIBGRAPI)*. IEEE, 346–353.
- [16] Serge Beucher. 1992. The watershed transformation applied to image segmentation. *Scanning Microscopy* 1992, 6 (1992), 28.
- [17] Serge Bobbia, Richard Macwan, Yannick Beneszeth, Keisuke Nakamura, Randy Gomez, and Julien Dubois. 2021. Iterative Boundaries implicit Identification for superpixels Segmentation: a real-time approach. *IEEE Access* 9 (2021), 77250–77263.
- [18] Isabela Borlido Barcelos, Felipe Belém, Paulo Miranda, Alexandre Xavier Falcão, Zenilton KG do Patrocínio, and Silvio Jamil F Guimarães. 2021. Towards interactive image segmentation by dynamic and iterative spanning forest. In *International Conference on Discrete Geometry and Mathematical Morphology*. Springer, 351–364.
- [19] Stephen Boyd, Neal Parikh, Eric Chu, Borja Peleato, and Jonathan Eckstein. 2011. Distributed Optimization and Statistical Learning via the Alternating Direction Method of Multipliers. *Foundations and Trends® in Machine Learning* 3, 1 (2011), 1–122. <https://doi.org/10.1561/2200000016>
- [20] Pierre Buyssens, Isabelle Gardin, and Su Ruan. 2014. Eikonal based region growing for superpixels generation: Application to semi-supervised real time organ segmentation in CT images. *IRBM* 35, 1 (2014), 20–26. <https://doi.org/10.1016/j.irbm.2013.12.007> Biomedical image segmentation using variational and statistical approaches.
- [21] Dengfeng Chai. 2020. Rooted Spanning Superpixels. *International Journal of Computer Vision* 128, 12 (2020), 2962–2978.
- [22] Jia-Ren Chang and Yong-Sheng Chen. 2018. Pyramid stereo matching network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 5410–5418.
- [23] Jiansheng Chen, Zhengqin Li, and Bo Huang. 2017. Linear Spectral Clustering Superpixel. *IEEE Transactions on Image Processing* 26, 7 (2017), 3317–3330. <https://doi.org/10.1109/TIP.2017.2651389>
- [24] Liang-Chieh Chen, George Papandreou, Florian Schroff, and Hartwig Adam. 2017. Rethinking atrous convolution for semantic image segmentation. *arXiv preprint arXiv:1706.05587* (2017).
- [25] Liang-Chieh Chen, Yukun Zhu, George Papandreou, Florian Schroff, and Hartwig Adam. 2018. Encoder-Decoder with Atrous Separable Convolution for Semantic Image Segmentation. In *Proceedings of the European Conference on Computer Vision (ECCV)*.
- [26] Marcos A.T. Condori, Fábio A.M. Cappabianco, Alexandre X. Falcão, and Paulo A.V. Miranda. 2020. An extension of the differential image foresting transform and its application to superpixel generation. *Journal of Visual Communication and Image Representation* 71 (2020), 102748. <https://doi.org/10.1016/j.jvcir.2019.102748>
- [27] Marcos Ademir Tejada Condori, Fabio Augusto Menocci Cappabianco, Alexandre Xavier Falcão, and Paulo Andre Vechiatto De Miranda. 2017. Extending the differential image foresting transform to root-based path-cost functions with application to superpixel segmentation. In *2017 30th SIBGRAPI Conference on Graphics, Patterns and Images (SIBGRAPI)*. Ieee, 7–14.
- [28] Christian Conrad, Matthias Mertz, and Rudolf Mester. 2013. Contour-Relaxed Superpixels. In *Energy Minimization Methods in Computer Vision and Pattern Recognition*. Springer Berlin Heidelberg, Berlin, Heidelberg, 280–293.
- [29] Andrew Delong, Anton Osokin, Hossam N Isack, and Yuri Boykov. 2012. Fast approximate energy minimization with label costs. *International journal of computer vision* 96, 1 (2012), 1–27.
- [30] Shuanhu Di, Miao Liao, Yuqian Zhao, Yang Li, and Yezhan Zeng. 2021. Image superpixel segmentation based on hierarchical multi-level LI-SLIC. *Optics & Laser Technology* 135 (2021), 106703.
- [31] Edsger W Dijkstra et al. 1959. A note on two problems in connexion with graphs. *Numerische mathematik* 1, 1 (1959), 269–271.
- [32] Moshe Eliasof, Nir Ben Zikri, and Eran Treister. 2022. Rethinking unsupervised neural superpixel segmentation. In *2022 IEEE International Conference on Image Processing (ICIP)*. IEEE, 3500–3504. <https://doi.org/10.1109/ICIP46576.2022.9897484>

- [33] Alexandre X Falcão and Felipe PG Bergo. 2004. Interactive volume segmentation with differential image foresting transforms. *IEEE Transactions on Medical Imaging* 23, 9 (2004), 1100–1108.
- [34] Alexandre X Falcão, Jorge Stolfi, and Roberto de Alencar Lotufo. 2004. The image foresting transform: Theory, algorithms, and applications. *IEEE transactions on pattern analysis and machine intelligence* 26, 1 (2004), 19–29.
- [35] Leyuan Fang, Shutao Li, Xudong Kang, and Jon Atli Benediktsson. 2015. Spectral–spatial classification of hyperspectral images with a superpixel-based discriminative sparse model. *IEEE Transactions on Geoscience and Remote Sensing* 53, 8 (2015), 4186–4201.
- [36] Jobin Francis, M Baburaj, and Sudhish N George. 2022. An  $l_{1/2}$  and Graph Regularized Subspace Clustering Method for Robust Image Segmentation. *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)* 18, 2 (2022), 1–24.
- [37] Felipe Lemes Galvão, Alexandre Xavier Falcão, and Ananda Shankar Chowdhury. 2018. RISF: recursive iterative spanning forest for superpixel segmentation. In *2018 31st SIBGRAPI Conference on Graphics, Patterns and Images (SIBGRAPI)*. IEEE, 408–415.
- [38] Felipe Lemes Galvão, Silvio Jamil Ferzoli Guimarães, and Alexandre Xavier Falcão. 2020. Image segmentation using dense and sparse hierarchies of superpixels. *Pattern Recognition* 108 (2020), 107532. <https://doi.org/10.1016/j.patcog.2020.107532>
- [39] Utkarsh Gaur and B. S. Manjunath. 2020. Superpixel Embedding Network. *IEEE Transactions on Image Processing* 29 (2020), 3199–3212. <https://doi.org/10.1109/TIP.2019.2957937>
- [40] Rémi Giraud, Vinh-Thong Ta, and Nicolas Papadakis. 2016. SCALP: Superpixels with Contour Adherence using Linear Path. In *2016 23rd International Conference on Pattern Recognition (ICPR)*. 2374–2379. <https://doi.org/10.1109/ICPR.2016.7899991>
- [41] Rémi Giraud, Vinh-Thong Ta, and Nicolas Papadakis. 2018. Robust superpixels using color and contour features along linear path. *Computer Vision and Image Understanding* 170 (2018), 1–13.
- [42] Clément Godard, Oisin Mac Aodha, and Gabriel J Brostow. 2017. Unsupervised monocular depth estimation with left-right consistency. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 270–279.
- [43] Jianglei Gong, Nannan Liao, Cheng Li, Xiaojun Ma, Wangpeng He, and Baolong Guo. 2021. Superpixel Segmentation via Contour Optimized Non-Iterative Clustering. In *International Conference on Neural Computing for Advanced Applications*. Springer, 645–658.
- [44] Yikai Gong, Richard O. Sinnott, and Paul Rimba. 2018. RT-DBSCAN: Real-Time Parallel Clustering of Spatio-Temporal Data Using Spark-Streaming. In *Computational Science – ICCS 2018*, Yong Shi, Haohuan Fu, Yingjie Tian, Valeria V. Krzhizhanovskaya, Michael Harold Lees, Jack Dongarra, and Peter M. A. Sloot (Eds.). Springer International Publishing, Cham, 524–539.
- [45] Leo Grady. 2006. Random walks for image segmentation. *IEEE transactions on pattern analysis and machine intelligence* 28, 11 (2006), 1768–1783.
- [46] Reaya Grewal, Singara Singh Kasana, and Geeta Kasana. 2023. Hyperspectral image segmentation: a comprehensive survey. *Multimedia Tools and Applications* 82, 14 (2023), 20819–20872. <https://doi.org/10.1007/s11042-022-13959-w>
- [47] Junyi Guan, Sheng Li, Xiongxiong He, and Jiajia Chen. 2021. Peak-Graph-Based Fast Density Peak Clustering for Image Segmentation. *IEEE Signal Processing Letters* 28 (2021), 897–901.
- [48] Alvaro Guevara, Christian Conrad, and Rudolf Mester. 2011. Boosting segmentation results by contour relaxation. In *2011 18th IEEE International Conference on Image Processing*. IEEE, 1405–1408.
- [49] Alvaro Guevara, Christian Conrad, and Rudolf Mester. 2011. Boosting segmentation results by contour relaxation. In *2011 18th IEEE International Conference on Image Processing*. IEEE, 1405–1408. <https://doi.org/10.1109/ICIP.2011.6115703>
- [50] Laurent Guigues, Jean Pierre Cocquerez, and Hervé Le Men. 2006. Scale-sets image analysis. *International Journal of Computer Vision* 68, 3 (2006), 289–317.
- [51] Ashish Kumar Gupta, Ayan Seal, Pritee Khanna, Ondrej Krejcar, and Anis Yazidi. 2021. AWkS: adaptive, weighted k-means-based superpixels for improved saliency detection. *Pattern Analysis and Applications* 24, 2 (2021), 625–639.
- [52] Xiaodi Hou and Liqing Zhang. 2007. Saliency Detection: A Spectral Residual Approach. In *2007 IEEE Conference on Computer Vision and Pattern Recognition*. 1–8. <https://doi.org/10.1109/CVPR.2007.383267>
- [53] Jie Hu, Li Shen, and Gang Sun. 2018. Squeeze-and-Excitation Networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 7132–7141.
- [54] Joshua Zhixue Huang, Michael K Ng, Hongqiang Rong, and Zichen Li. 2005. Automated variable weighting in k-means type clustering. *IEEE transactions on pattern analysis and machine intelligence* 27, 5 (2005), 657–668.
- [55] Varun Jampani, Deqing Sun, Ming-Yu Liu, Ming-Hsuan Yang, and Jan Kautz. 2018. Superpixel sampling networks. In *Proceedings of the European Conference on Computer Vision (ECCV)*. 352–368.
- [56] Zexuan Ji, Yubo Huang, Quansen Sun, and Guo Cao. 2016. A spatially constrained generative asymmetric Gaussian mixture model for image segmentation. *Journal of Visual Communication and Image Representation* 40 (2016), 611–626.

- [57] Xuejing Kang, Lei Zhu, and Anlong Ming. 2020. Dynamic Random Walk for Superpixel Segmentation. *IEEE Transactions on Image Processing* 29 (2020), 3871–3884. <https://doi.org/10.1109/TIP.2020.2967583>
- [58] Song Ke-Chen, YAN Yun-Hui, CHEN Wen-Hui, and Xu Zhang. 2013. Research and perspective on local binary pattern. *Acta Automatica Sinica* 39, 6 (2013), 730–744.
- [59] Shu Kong and Charless C Fowlkes. 2018. Recurrent pixel embedding for instance grouping. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 9018–9028.
- [60] Andreas Krause, Pietro Perona, and Ryan Gomes. 2010. Discriminative clustering by regularized information maximization. *Advances in neural information processing systems* 23 (2010), 775–783.
- [61] B Vinoth Kumar. 2023. An Extensive Survey on Superpixel Segmentation: A Research Perspective. *Archives of Computational Methods in Engineering* (2023), 1–19. <https://doi.org/10.1007/s11831-023-09919-8>
- [62] Suha Kwak, Seunghoon Hong, and Bohyung Han. 2017. Weakly supervised semantic segmentation using superpixel pooling network. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 31. 4111–4117.
- [63] Victor Lempitsky, Andrea Vedaldi, and Dmitry Ulyanov. 2018. Deep image prior. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*. IEEE, 9446–9454.
- [64] Alex Levinstein, Adrian Stere, Kiriakos N Kutulakos, David J Fleet, Sven J Dickinson, and Kaleem Siddiqi. 2009. Turbopixels: Fast superpixels using geometric flows. *IEEE transactions on pattern analysis and machine intelligence* 31, 12 (2009), 2290–2297.
- [65] Hua Li, Yuheng Jia, Runmin Cong, Wenhui Wu, Sam Tak Wu Kwong, and Chuanbo Chen. 2020. Superpixel segmentation based on spatially constrained subspace clustering. *IEEE Transactions on Industrial Informatics* 17, 11 (2020), 7501–7512.
- [66] Xiangjun Li, Yong Zhou, Xinpeng Zhang, Su Xu, and Peng Yu. 2021. Cluster-based fine-to-coarse superpixel segmentation. *Engineering Applications of Artificial Intelligence* 102 (2021), 104281.
- [67] Zhengqin Li and Jiansheng Chen. 2015. Superpixel Segmentation Using Linear Spectral Clustering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 1356–1363.
- [68] Yun Liang, Yaqing Zhang, Yihan Wu, Shujin Tu, and Caixing Liu. 2020. Robust video object segmentation via propagating seams and matching superpixels. *IEEE Access* 8 (2020), 53766–53776.
- [69] Guohua Liu and Jianchun Duan. 2020. RGB-D image segmentation using superpixel and multi-feature fusion graph theory. *Signal, Image and Video Processing* 14, 6 (2020), 1171–1179.
- [70] Jiafei Liu, Qingsong Wang, Jianda Cheng, Deliang Xiang, and Wenbo Jing. 2022. Multitask Learning-Based for SAR Image Superpixel Generation. *Remote Sensing* 14, 4 (2022). <https://doi.org/10.3390/rs14040899>
- [71] Ming-Yu Liu, Oncel Tuzel, Srikanth Ramalingam, and Rama Chellappa. 2011. Entropy rate superpixel segmentation. In *CVPR 2011*. 2097–2104. <https://doi.org/10.1109/CVPR.2011.5995323>
- [72] Yun Liu, Ming-Ming Cheng, Xiaowei Hu, Kai Wang, and Xiang Bai. 2017. Richer Convolutional Features for Edge Detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.
- [73] Yun Liu, Peng-Tao Jiang, Vahan Petrosyan, Shi-Jie Li, Jiawang Bian, Le Zhang, and Ming-Ming Cheng. 2018. DEL: Deep Embedding Learning for Efficient Image Segmentation. In *IJCAI*. 864–870. [10.24963/ijcai.2018/120](https://doi.org/10.24963/ijcai.2018/120)
- [74] Seng Cheong Loke, Bruce A MacDonald, Matthew Parsons, and Burkhard Claus Wünsche. 2021. Accelerated superpixel image segmentation with a parallelized DBSCAN algorithm. *Journal of Real-Time Image Processing* 18, 6 (2021), 2361–2376.
- [75] Can-Yi Lu, Hai Min, Zhong-Qiu Zhao, Lin Zhu, De-Shuang Huang, and Shuicheng Yan. 2012. Robust and efficient subspace segmentation via least squares regression. In *European conference on computer vision*. Springer, 347–360.
- [76] Dongyang Ma, Yuanfeng Zhou, Shiqing Xin, and Wenping Wang. 2020. Convex and compact superpixels by edge-constrained centroidal power diagram. *IEEE Transactions on Image Processing* 30 (2020), 1825–1839.
- [77] Vaïa Machairas, Matthieu Faessel, David Cárdenas-Peña, Théodore Chabardes, Thomas Walter, and Etienne Decenciere. 2015. Waterpixels. *IEEE Transactions on Image Processing* 24, 11 (2015), 3707–3716.
- [78] Georg Maierhofer, Daniel Heydecker, Angelica I. Aviles-Rivero, Samar M. Alsaleh, and Carola-Bibiane Schonlieb. 2018. Peekaboo-Where are the Objects? Structure Adjusting Superpixels. In *2018 25th IEEE International Conference on Image Processing (ICIP)*. 3693–3697. <https://doi.org/10.1109/ICIP.2018.8451822>
- [79] Lucy AC Mansilla and Paulo AV Miranda. 2013. Image segmentation by oriented image foresting transform: Handling ties and colored images. In *2013 18th International Conference on Digital Signal Processing (DSP)*. IEEE, 1–6.
- [80] Lucy A. C. Mansilla and Paulo A. V. Miranda. 2016. Oriented Image Foresting Transform Segmentation: Connectivity Constraints with Adjustable Width. In *2016 29th SIBGRAPI Conference on Graphics, Patterns and Images (SIBGRAPI)*. 289–296. <https://doi.org/10.1109/SIBGRAPI.2016.047>
- [81] David R Martin, Charless C Fowlkes, and Jitendra Malik. 2004. Learning to detect natural image boundaries using local brightness, color, and texture cues. *IEEE transactions on pattern analysis and machine intelligence* 26, 5 (2004), 530–549. <https://doi.org/10.1109/TPAMI.2004.1273918>

- [82] Bérengère Mathieu, Alain Crouzil, and Jean Baptiste Puel. 2017. Oversegmentation methods: a new evaluation. In *Iberian Conference on Pattern Recognition and Image Analysis*. Springer, 185–193.
- [83] Rudolf Mester, Christian Conrad, and Alvaro Guevara. 2011. Multichannel segmentation using contour relaxation: fast super-pixels and temporal propagation. In *Scandinavian conference on image analysis*. Springer, 250–261.
- [84] Rudolf Mester, Christian Conrad, and Alvaro Guevara. 2011. Multichannel segmentation using contour relaxation: fast super-pixels and temporal propagation. In *Scandinavian conference on image analysis*, Anders Heyden and Fredrik Kahl (Eds.). Springer, Berlin, Heidelberg, 250–261.
- [85] Paulo AV Miranda and Lucy AC Mansilla. 2013. Oriented image foresting transform segmentation by seed competition. *IEEE Transactions on Image Processing* 23, 1 (2013), 389–398.
- [86] Alastair P Moore, Simon JD Prince, Jonathan Warrell, Umar Mohammed, and Graham Jones. 2008. Superpixel lattices. In *2008 IEEE conference on computer vision and pattern recognition*. IEEE, 1–8.
- [87] Peer Neubert and Peter Protzel. 2012. Superpixel benchmark and comparison. In *Proc. Forum Bildverarbeitung*, Vol. 6. KIT Scientific Publishing, 1–12.
- [88] Xiao Pan, Yuanfeng Zhou, Yunfeng Zhang, and Caiming Zhang. 2022. Fast Generation of Superpixels With Lattice Topology. *IEEE Transactions on Image Processing* 31 (2022), 4828–4841. <https://doi.org/10.1109/TIP.2022.3188155>
- [89] Hankui Peng, Angelica I Aviles-Rivero, and Carola-Bibiane Schönlieb. 2022. HERs Superpixels: Deep Affinity Learning for Hierarchical Entropy Rate Segmentation. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*. 217–226.
- [90] George Polya. 2020. *Mathematics and plausible reasoning: Induction and analogy in mathematics*. Vol. 1. Princeton University Press.
- [91] Nianzu Qiao and Lamei Di. 2022. An improved method of linear spectral clustering. *Multimedia Tools and Applications* 81, 1 (2022), 1287–1311.
- [92] Xuebin Qin, Zichen Zhang, Chenyang Huang, Masood Dehghan, Osmar R Zaiane, and Martin Jagersand. 2020. U2-Net: Going deeper with nested U-structure for salient object detection. *Pattern Recognition* 106 (2020), 107404.
- [93] Lang Ren, Liaoying Zhao, and Yulei Wang. 2019. A superpixel-based dual window RX for hyperspectral anomaly detection. *IEEE Geoscience and Remote Sensing Letters* 17, 7 (2019), 1233–1237.
- [94] Xiaofeng Ren and Jitendra Malik. 2003. Learning a classification model for segmentation. In *Proceedings Ninth IEEE International Conference on Computer Vision*, Vol. 2. IEEE Computer Society, 10–17. <https://doi.org/10.1109/ICCV.2003.1238308>
- [95] Alex Rodriguez and Alessandro Laio. 2014. Clustering by fast search and find of density peaks. *Science* 344, 6191 (2014), 1492–1496. <https://doi.org/10.1126/science.1242072>
- [96] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. 2015. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*. Springer, 234–241.
- [97] Buddhadev Sasmal and Krishna Gopal Dhal. 2023. A survey on the utilization of Superpixel image for clustering based image segmentation. *Multimedia Tools and Applications* (2023), 1–63. <https://doi.org/10.1007/s11042-023-14861-9>
- [98] Alexander Schick, Mika Fischer, and Rainer Stiefelhagen. 2012. Measuring and evaluating the compactness of superpixels. In *Proceedings of the 21st international conference on pattern recognition (ICPR2012)*. IEEE, 930–934.
- [99] Alexander Schick, Mika Fischer, and Rainer Stiefelhagen. 2014. An evaluation of the compactness of superpixels. *Pattern Recognition Letters* 43 (2014), 71–80.
- [100] Philip Sellars, Angelica I Aviles-Rivero, and Carola-Bibiane Schönlieb. 2020. Superpixel contracted graph-based learning for hyperspectral image classification. *IEEE Transactions on Geoscience and Remote Sensing* 58, 6 (2020), 4180–4193.
- [101] James Albert Sethian. 1999. *Level set methods and fast marching methods: evolving interfaces in computational geometry, fluid mechanics, computer vision, and materials science*. Vol. 3. Cambridge university press.
- [102] Sayed Asad Hussain Shah, Liang Li, Yajun Li, and Jiawan Zhang. 2021. Superpixel Segmentation via Density Peaks. In *2021 4th International Conference on Information and Computer Technologies (ICICT)*. IEEE, 93–98.
- [103] Jianbing Shen, Xiaopeng Hao, Zhiyuan Liang, Yu Liu, Wenguan Wang, and Ling Shao. 2016. Real-Time Superpixel Segmentation by DBSCAN Clustering Algorithm. *IEEE Transactions on Image Processing* 25, 12 (2016), 5933–5942. <https://doi.org/10.1109/TIP.2016.2616302>
- [104] Bin Sheng, Ping Li, Shuangjia Mo, Huating Li, Xuhong Hou, Qiang Wu, Jing Qin, Ruogu Fang, and David Dagan Feng. 2018. Retinal vessel segmentation using minimum spanning superpixel tree detector. *IEEE transactions on cybernetics* 49, 7 (2018), 2707–2719.
- [105] Jianbo Shi and J. Malik. 2000. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 22, 8 (2000), 888–905. <https://doi.org/10.1109/34.868688>
- [106] Jianping Shi, Qiong Yan, Li Xu, and Jiaya Jia. 2015. Hierarchical image saliency detection on extended CSSD. *IEEE transactions on pattern analysis and machine intelligence* 38, 4 (2015), 717–729.

- [107] Nathan Silberman, Derek Hoiem, Pushmeet Kohli, and Rob Fergus. 2012. Indoor segmentation and support inference from RGB-D images. In *Computer Vision–ECCV 2012: 12th European Conference on Computer Vision, Florence, Italy, October 7–13, 2012, Proceedings, Part V 12*. Springer, 746–760.
- [108] David Stutz. 2015. Superpixel segmentation: An evaluation. In *German conference on pattern recognition*. Springer, 555–562.
- [109] David Stutz, Alexander Hermans, and Bastian Leibe. 2018. Superpixels: An evaluation of the state-of-the-art. *Computer Vision and Image Understanding* 166 (2018), 1–27.
- [110] Teppei Suzuki. 2020. Superpixel Segmentation Via Convolutional Neural Networks with Regularized Information Maximization. In *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2573–2577. <https://doi.org/10.1109/ICASSP40776.2020.9054140>
- [111] Carlo Tomasi and Roberto Manduchi. 1998. Bilateral filtering for gray and color images. In *Sixth international conference on computer vision (IEEE Cat. No. 98CH36271)*. IEEE, 839–846.
- [112] Wei-Chih Tu, Ming-Yu Liu, Varun Jampani, Deqing Sun, Shao-Yi Chien, Ming-Hsuan Yang, and Jan Kautz. 2018. Learning superpixels with segmentation-aware affinity loss. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 568–576.
- [113] Shakir Ullah, Naeem Bhatti, and Muhammad Zia. 2021. Adaptive tuning of SLIC parameter K. *Multimedia Tools and Applications* 80, 17 (2021), 25649–25672.
- [114] Michael Van den Bergh, Xavier Boix, Gemma Roig, Benjamin de Capitani, and Luc Van Gool. 2012. SEEDS: Superpixels Extracted via Energy-Driven Sampling. In *Computer Vision – ECCV 2012*, Andrew Fitzgibbon, Svetlana Lazebnik, Pietro Perona, Yoichi Sato, and Cordelia Schmid (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 13–26.
- [115] Michael Van den Bergh, Xavier Boix, Gemma Roig, and Luc Van Gool. 2015. Seeds: Superpixels extracted via energy-driven sampling. *International Journal of Computer Vision* 111, 3 (2015), 298–314. <https://doi.org/10.1007/s11263-014-0744-2>
- [116] John E Vargas-Muñoz, Ananda S Chowdhury, Eduardo B Alexandre, Felipe L Galvão, Paulo A Vechiatto Miranda, and Alexandre X Falcão. 2019. An iterative spanning forest framework for superpixel segmentation. *IEEE Transactions on Image Processing* 28, 7 (2019), 3477–3489.
- [117] Gaochao Wang, Yiheng Wei, and Peter Tse. 2018. Clustering by defining and merging candidates of cluster centers via independence and affinity. *Neurocomputing* 315 (2018), 486–495. <https://doi.org/10.1016/j.neucom.2018.07.043>
- [118] Hui Wang, Jianbing Shen, Junbo Yin, Xingping Dong, Hanqiu Sun, and Ling Shao. 2020. Adaptive Nonlocal Random Walks for Image Superpixel Segmentation. *IEEE Transactions on Circuits and Systems for Video Technology* 30, 3 (2020), 822–834. <https://doi.org/10.1109/TCSVT.2019.2896438>
- [119] Jingjing Wang, Zhenye Luan, Zishu Yu, Jinwen Ren, Jun Gao, Kejiang Yuan, and Huaqiang Xu. 2022. Superpixel segmentation with squeeze-and-excitation networks. *Signal, Image and Video Processing* (2022), 1–8.
- [120] Kai Wang, Liang Li, and Jiawan Zhang. 2020. End-to-end trainable network for superpixel and image segmentation. *Pattern Recognition Letters* 140 (2020), 135–142. <https://doi.org/10.1016/j.patrec.2020.09.016>
- [121] Longguang Wang, Yingqian Wang, Zhengfa Liang, Zaiping Lin, Jungang Yang, Wei An, and Yulan Guo. 2019. Learning Parallax Attention for Stereo Image Super-Resolution. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.
- [122] Murong Wang, Xiabi Liu, Yixuan Gao, Xiao Ma, and Nouman Q Soomro. 2017. Superpixel segmentation: A benchmark. *Signal Processing: Image Communication* 56 (2017), 28–39.
- [123] Nannan Wang and Yongxia Zhang. 2021. Adaptive and fast image superpixel segmentation approach. *Image and Vision Computing* 116 (2021), 104315.
- [124] Weiwei Wang and Cuiling Wu. 2017. Image segmentation by correlation adaptive weighted regression. *Neurocomputing* 267 (2017), 426–435. <https://doi.org/10.1016/j.neucom.2017.06.046>
- [125] Xuehui Wang, Qingyun Zhao, Lei Fan, Yuzhi Zhao, Tiantian Wang, Qiong Yan, and Long Chen. 2021. Semasuperpixel: A Multi-Channel Probability-Driven Superpixel Segmentation Method. In *2021 IEEE International Conference on Image Processing (ICIP)*. IEEE, 1859–1863.
- [126] Yaxiong Wang, Yunchao Wei, Xueming Qian, Li Zhu, and Yi Yang. 2021. AI-Net: Association implantation for superpixel segmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 7078–7087.
- [127] Xing Wei, Qingxiong Yang, Yihong Gong, Narendra Ahuja, and Ming-Hsuan Yang. 2018. Superpixel hierarchy. *IEEE Transactions on Image Processing* 27, 10 (2018), 4838–4849.
- [128] Douglas Brent West et al. 2001. *Introduction to graph theory*. Vol. 2. Prentice hall Upper Saddle River.
- [129] Jiang Wu, Chunxiao Liu, and Biao Li. 2021. Texture-aware and structure-preserving superpixel segmentation. *Computers & Graphics* 94 (2021), 152–163.
- [130] Ruiqi Wu, Yajuan Du, Hua Li, and Yucong Dai. 2021. Stereo Superpixel Segmentation Via Dual-Attention Fusion Networks. In *2021 IEEE International Conference on Multimedia and Expo (ICME)*. IEEE, 1–6.

- [131] Xiang Wu, Yufei Chen, Xianhui Liu, Jianan Shen, Keqiang Zhuo, and Weidong Zhao. 2020. Superpixel via coarse-to-fine boundary shift. *Applied Intelligence* 50, 7 (2020), 2079–2092.
- [132] Juanying Xie, Hongchao Gao, Weixin Xie, Xiaohui Liu, and Philip W. Grant. 2016. Robust clustering by detecting density peaks and assigning points based on fuzzy weighted K-nearest neighbors. *Information Sciences* 354 (2016), 19–40. <https://doi.org/10.1016/j.ins.2016.03.011>
- [133] Koichiro Yamaguchi, David McAllester, and Raquel Urtasun. 2014. Efficient joint segmentation, occlusion labeling, stereo and flow estimation. In *European Conference on Computer Vision*, Fleet David, Tomas Pajdla, Bernt Schiele, and Tinne Tuytelaars (Eds.), Vol. 8693. Springer, Cham, 756–771. [https://doi.org/10.1007/978-3-319-10602-1\\_49](https://doi.org/10.1007/978-3-319-10602-1_49)
- [134] Fengting Yang, Qian Sun, Hailin Jin, and Zihan Zhou. 2020. Superpixel Segmentation With Fully Convolutional Networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.
- [135] Jian Yao, Marko Boben, Sanja Fidler, and Raquel Urtasun. 2015. Real-time coarse-to-fine topologically preserving segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2947–2955.
- [136] Yue Yu, Yang Yang, and Kezhao Liu. 2021. Edge-Aware Superpixel Segmentation with Unsupervised Convolutional Neural Networks. In *2021 IEEE International Conference on Image Processing (ICIP)*. IEEE, 1504–1508.
- [137] Hongxing Yuan, Shaoqun Wu, Peihong Cheng, Peng An, and Shudi Bao. 2015. Nonlocal Random Walks Algorithm for Semi-Automatic 2D-to-3D Image Conversion. *IEEE Signal Processing Letters* 22, 3 (2015), 371–374. <https://doi.org/10.1109/LSP.2014.2359643>
- [138] Qing Yuan, Songfeng Lu, Yan Huang, and Wuxin Sha. 2021. SIN: Superpixel Interpolation Network. In *Pacific Rim International Conference on Artificial Intelligence*. Springer, 293–307.
- [139] Ye Yuan, Wei Zhang, Hai Yu, and Zhiliang Zhu. 2021. Superpixels With Content-Adaptive Criteria. *IEEE Transactions on Image Processing* 30 (2021), 7702–7716.
- [140] Ye Yuan, Zhiliang Zhu, Hai Yu, and Wei Zhang. 2020. Watershed-Based Superpixels With Global and Local Boundary Marching. *IEEE Transactions on Image Processing* 29 (2020), 7375–7388. <https://doi.org/10.1109/TIP.2020.3002078>
- [141] Bin Zhang, Xuejing Kang, and Anlong Ming. 2021. BP-net: deep learning-based superpixel segmentation for RGB-D image. In *2020 25th International Conference on Pattern Recognition (ICPR)*. IEEE, 7433–7438.
- [142] Jianchao Zhang, Angelica I Aviles-Rivero, Daniel Heydecker, Xiaosheng Zhuang, Raymond Chan, and Carola-Bibiane Schönlieb. 2021. Dynamic spectral residual superpixels. *Pattern Recognition* 112 (2021), 107705.
- [143] Jianhua Zhang, Jingbo Chen, Qichao Wang, and Shengyong Chen. 2019. Spatiotemporal saliency detection based on maximum consistency superpixels merging for video analysis. *IEEE Transactions on Industrial Informatics* 16, 1 (2019), 606–614.
- [144] Yongxia Zhang, Qiang Guo, and Caiming Zhang. 2021. Simple and fast image superpixels generation with color and boundary probability. *The Visual Computer* 37 (2021), 1061–1074. <https://doi.org/10.1007/s00371-020-01852-2>
- [145] Wei Zhao, Yi Fu, Xiaosong Wei, and Hai Wang. 2018. An improved image semantic segmentation method based on superpixels and conditional random fields. *Applied Sciences* 8, 5 (2018), 837.
- [146] Xianen Zhou, Yaonan Wang, Qing Zhu, Changyan Xiao, and Xiao Lu. 2019. SSG: superpixel segmentation and grabcut-based salient object segmentation. *The Visual Computer* 35, 3 (2019), 385–398.
- [147] Lei Zhu, Qi She, Bin Zhang, Yanye Lu, Zhilin Lu, Duo Li, and Jie Hu. 2021. Learning the Superpixel in a Non-iterative and Lifelong Manner. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 1225–1234.
- [148] Fariba Zohrizadeh, Mohsen Kheirandishfard, and Farhad Kamangar. 2018. Image Segmentation Using Sparse Subset Selection. In *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*. 1470–1479. <https://doi.org/10.1109/WACV.2018.00165>

## A SUPERPIXEL SEGMENTATION METHODS

Superpixel segmentation has a vast literature covering several techniques. In [109], a benchmark for superpixels is provided with an extensive evaluation of methods. Nevertheless, due to the rapid progress in developing new strategies for superpixel segmentation, an analysis of the most recent proposals becomes essential. Therefore, this section reviews 59 methods in recent and commonly used literature on superpixel segmentation.

### A.1 Neighborhood-based clustering

Neighborhood-based methods for superpixel segmentation perform clustering of image pixels based on the similarity between pixels restricted to a maximum spatial distance from some reference

point in the image. For example, several methods constrain the clustering region of a superpixel to a fixed-size image patch around this superpixel [1, 69, 113, 129].

**A.1.1 SLIC.** SLIC [1] starts with a grid sampling of superpixel centers and iteratively assigns at each superpixel the most similar pixels in a limited region around the superpixel center. As post-processing, SLIC ensures connectivity by assigning unconnected superpixels to their nearest neighbors. SLIC reduces the segmentation complexity to linear concerning the number of pixels. Also, its distance function gives better control over the superpixel size and compactness. Although SLIC presents fair delineation and efficiency, it does not consider the relationship between adjacent pixels, resulting in worse delineation in regions with complex textures.

**A.1.2 K-SLIC.** The authors [113] propose a granulometric approach and a quality metric method to allow controlling the number of desired superpixels in a SLIC [1] segmentation. The former represents the relative importance of the image components computed for each color channel and the second uses several metrics based on entropy, texture, and ground-truth independent quality metrics to choose by the majority vote. In bad-lighted conditions, the quality metric method is less affected and provides a large number of superpixels as compared to the granulometric process and performs better with different spatial resolutions. Despite its improved results, the quality metric method is computationally expensive, while the granulometric one has worse performance.

**A.1.3 LSC.** The authors [23, 67] investigated the relationship between the normalized cuts [94] and the weighted K-means to propose the LSC, which uses an NCut function that can obtain the same optimum result as the weighted kernel K-means. The LSC applies a kernel function to map pixels into a 10-dimensional feature space in a fixed limited region. LSC provides an efficient segmentation method and it obtains regular shapes. It also has linear time complexity with high memory efficiency. By considering a shape constraint, LSC achieves high boundary adherence without sacrificing spatial compactness. However, its fixed search range prevents LSC from ensuring connectivity, requiring post-processing.

**A.1.4 SCALP.** SCALP [40] considers image features and contour intensity on a linear path to the superpixel barycenter to improve SLIC's [1] distance function with neighborhood information. It integrates the contour prior information as a soft constraint in the color distance to improve the adherence to the object boundaries and performs clustering in high-dimensional feature space [67]. SCALP is efficient, robust to noise, and produces compact superpixels. The authors further improve SCALP [41] with a hard constraint based on the contour prior to providing an initial segmentation. The hard constraint increases SCALP's robustness and its boundary adherence, but it slightly reduces regularity and smoothness.

**A.1.5 TASP.** TASP [129] intends to solve the problem of handling weak gradient structures and strong gradient textures. The proposal's pipeline is based on SLIC [1] with an integrated structure-avoiding clustering distance based on a centroid-oriented quarter-circular mask and a hybrid gradient. The proposed mask prevents inconsistent texture pixels from being sampled from the local image patch. TASP has an effective structure-preserving and texture-suppression procedure, especially in images with strong texture and weak boundary structures. However, TASP is highly time-consuming and does not produce more superpixels in regions with finer details, missing some structure boundaries.

**A.1.6 MFGS.** MFGS [69] is a two-stage method for superpixel segmentation in RGB-D images. In the first stage, MFGS uses color, and 2D and 3D spatial positions (with depth) to perform iterative clustering. Then, it performs a merging multi-feature step to estimate the similarity between superpixels. Also, it uses the label cost proposed in [29] to remove redundant labels. MFGS is faster,

produces compact and regular superpixels, and has a higher segmentation accuracy. However, the proposal's merging stage does not allow control of the number of final superpixels.

**A.1.7 DSR.** DSR [142] incorporates saliency information into the seed sampling and clustering stages. The proposed method computes the saliency map based on Fourier analysis [52] and uses a structure measure function to define the search range for clustering and seed sampling. DSR performs clustering similar to SLIC [1], but with a search range based on the structure measure to connect uniform regions, avoiding unnecessary small superpixels in large regions. DSR creates more seeds in heterogeneous areas but avoids creating redundant seeds. Also, it produces larger (and fewer) superpixels on homogenous regions, by connecting pixels in a range search based on saliency. Compared to SLIC, DSR provides a consistent performance improvement by increasing a low computational load, producing superpixels that capture more details, and reducing the redundancy of the represented information. However, it creates less regular and compact superpixels.

**A.1.8 Semasuperpixel.** The proposal in [125] improves SLIC [1] clustering with a new distance measure function including semantic information. The proposal clusters pixels based on semantic information and uses color and spatial information as refinement factors. The authors use a DeepLab v3+ [25] network to obtain semantic information. Semasuperpixel achieves excellent boundary adherence and substantially reduces leakage achieving improved performance compared to SLIC.

**A.1.9 AWkS.** AWkS [51] adopts dynamic weighted distances based on weighted k-means clustering (W-k-means) [54] and proposes an adaptative term for each variable in its distance formulation. The proposed method extends SLIC [1] to explore the degree of feature relevances during objective function minimization, adopting a pipeline similar to SLIC AWks outperforms SLIC in boundary adherence and produces visually better segmentations, with more compact superpixels and fewer small ones close to the image boundaries. However, the proposal has a high running time compared to SLIC.

## A.2 Boundary evolution clustering

In boundary evolution clustering, the algorithm iteratively updates the superpixels' boundaries to improve delineation, usually using a coarse-to-fine image block strategy. SEEDS [114] and ETPS [135] are examples of superpixel methods using the boundary evolution strategy for clustering.

**A.2.1 SEEDS.** SEEDS [115] start from a regular grid partitioning and iteratively refine the superpixels' boundaries. The iterative process follows a coarse-to-fine approach with a hill-climbing algorithm for optimization. SEEDS is an efficient method that performs optimization based on a hill-climbing algorithm. SEEDS introduces an energy function that encourages color homogeneity, shape regularity, and smooth boundary shapes. However, the compactness constraint degrades the results, and the number of superpixels is challenging to control.

**A.2.2 CRS.** CRS [28] formulates the segmentation problem as an estimation task and transforms the model in [49, 84] into a superpixel approach. From an initial image partition, CRS generates superpixels under the constraint of maximum homogeneity inside each image patch and maximum accordance of the contours with both the image content and a Gibbs-Markov random field model. CRS explicitly models the superpixel's shape and content as a statistical model, allowing it to handle an arbitrary number of feature channels. In addition, CRS allows direct control of the number of superpixels and their compacity.

**A.2.3 ETPS.** Inspired by SEEDS [114], ETPS [135] performs a coarse-to-fine approach to superpixel segmentation, starting with grid partitioning. ETPS uses a priority list to optimize its energy function. Also, despite its energy function being at the pixel level, it measures shape regularization,

color homogeneity, and smoothness of the contours. In addition, ETPS enforces connectivity and minimum size during the optimization process. The authors also presented a stereo version of the proposal and demonstrated that ETPS' efficiency surpasses SLIC [1]. Compared to [133], ETPS achieves a better convergence value in a single iteration.

**A.2.4 IBIS.** IBIS [17] starts with a grid segmentation and, using the same distance measure in SLIC [1] compares the pixels located on the edge of the blocks, subdividing them into four blocks assigned to another superpixel. At each iteration, pixels in non-homogeneous blocks are assigned to the nearest superpixel according to the SLIC's distance measure. After the clustering step, IBIS performs the same merging stage as SLIC. In [17], the authors also present a GPU variant aimed at real-time use cases, the IBIScuda. IBIS is faster than SLIC with similar boundary adherence. Also, its Cuda version can improve efficiency, reducing computational time.

**A.2.5 CFBS.** CFBS [131] aims to overcome the two main limitations of many methods based on k-means: redundancy and the need for post-processing. The proposal performs a coarse-to-fine pixel block optimization using an optimization function similar to SLIC [1]. The CFBS updates all pixel blocks in the superpixels' boundary while the centers are updated dynamically. The number of iterations is defined by the maximum split operations of the initial block pixels. The authors demonstrated the proposal's ability to increase the performance of k-means-based methods while reducing its running time for superpixel segmentation, along with different applications. However, the CFBS segmentation does not capture finer details in more complex image regions, leading to a worse adherence to the image borders in these regions.

**A.2.6 SCAC.** From an initial grid segmentation, SCAC [139] performs an accuracy step followed by a compactness step. The former relabels the superpixel boundaries to maximize the adherence to the object contours according to balanced color weighted and spatial distances. Then, the compactness step performs a second relabeling based on color, gradient, and texture filters to detect regions with meaningless content. The gradient, color, and texture filters identify homogeneous, noised, and similar texture pattern regions. SCAC identifies meaningless-content regions, produces more compact superpixels, and prioritizes accuracy on regions with meaningful content. The proposal can run in real-time, but its runtime increases with the number of superpixels. Also, SCAC provides limited control over the number of superpixels, producing a number similar to the desired.

**A.2.7 LSC-Manhattan.** LSC-Manhattan [91] improves LSC [23] performance with a distance measurement based on non-convex image features and Manhattan distance. The proposal classifies the input image according to its texture complexity for subsampling and performs a semantic segmentation using DeepLabV3+ [25] to classify whether a pixel is part of some convex region. The subsampling strategy labels pixels according to texture complexity, applying different subsampling ratios according to the texture complexity level. The LSC-Manhattan produces better segmentation than LSC, with a reduced running time. However, the proposed distance measure is based on a specific dataset, which can lead to generalization issues.

**A.2.8 FLS.** In [88] the authors proposed a superpixel approach focused on lattice topology consistency. The proposed *Fast Lattice Superpixels* (FLS) formulates the superpixel generation problem as an energy function optimized through a hill-climbing optimization algorithm constrained to maintain lattice topology. Using a multilevel block strategy similar to SEEDS [115], FLS adjusts the superpixel affiliation of each block in the superpixel boundary, processing each block at least once per level. In the last level (the pixel level), several iterations of pixel updating are performed to improve boundary delineation. Furthermore, efficiency improves due to the parallelization of

non-neighbor blocks. Instead of using hand-crafted features, FLS inputs features from a convolutional network based on SSN [55] that includes in its loss function the local similarity of pixels with their neighboring pixels. FLS maintains the lattice topology (*i.e.*, a fixed number of neighbor superpixels), and the local similarity loss function improves boundary delineation. However, it has low compactness, and the proposed network, like SSN, is not able to produce the exact number of superpixels.

### A.3 Dynamic-center-update clustering

The dynamic-center-update algorithms perform clustering with a distance function based on the features of the clusters, dynamically updating its centers. Unlike neighborhood-based clustering, this approach does not perform a limited regional search to calculate distances.

**A.3.1 SNIC.** SNIC [2] intends to overcome SLIC's limitations [1]. The proposal starts with a sampling grid and dynamically updates the centroids during the clustering process. Furthermore, instead of searching limited to an image patch, SNIC uses a priority queue to group neighboring pixels, similar to path-based approaches, but with a distance function based on the superpixel centroid. Due to its clustering process based on neighboring pixels, SNIC enforces connectivity without requiring post-processing. Furthermore, SNIC requires less memory and is computationally more efficient than SLIC. The authors also proposed an algorithm for polygonal segmentation called SNICPOLY, which starts with superpixels generated with SNIC.

**A.3.2 FCSS.** The proposal [66] uses an SNIC-based algorithm [2] with depth information. FCSS controls the clustering process with a priority queue, a distance function, and a color threshold. When the queue is empty, FCSS performs a relocation process to solve the miss segmentation problem caused by the initial seed position. During relocation, FCSS pushes new cluster centers to the queue and updates the color threshold. Finally, the proposal merges unconnected pixels. The FCSS is relatively fast, even with the addition of time complexity due to the seed relocation processing. Also, it achieves a visually balanced segmentation between compactness and boundary adherence. However, the FCSS segmentation does not capture finer details in structure-rich regions, even reducing the compactness factor.

**A.3.3 CONIC.** Based on SNIC [2] and SCALP [41], CONIC [43] incorporates contour prior in a new distance measure, named joint color-spatial-contour measurement, which prevents the boundary pixels from being assigned prematurely. The proposal achieves competitive performance compared to SNIC and SCALP, with moderate compactness and an improved F-measure and boundary precision. CONIC's superpixels have low sensitivity to the gradient variation in textured regions, leading to less boundary degradation. Compared to SNIC, CONIC avoids redundant feature distance computations and has faster execution. However, the contour prior fails to identify some weak image boundaries.

**A.3.4 SCBP.** SCBP [144] is a two-stage and non-iterative method based on DBSCAN [103]. In the first stage, SCBP clusters the pixels in the conventional image order with an adaptative distance measure, processing each pixel only once and dynamically updating the cluster centers. The adaptative distance measure weights the spatial and color distances, balanced by a boundary probability term computed with the Sobel operator. The second stage merges superpixels based on their combined size according to the expected superpixel size. The proposed method produces compact and regular superpixels in homogenous image regions and superpixels closer to the boundaries in complex regions. Therefore, SCBP has  $O(n)$  time complexity, with a running time close to DBSCAN.

**A.3.5 A-DBSCAN.** The proposal [123] adopts an adaptative threshold and uses a new distance measurement that constrains superpixel shapes based on the linear path from a pixel to a seed. The proposal also uses a local binary pattern (LBP) operator [58] to compute texture and manage the regularity and boundary adherence tradeoff. After the clustering step, the A-DBSCAN performs a merging stage to produce final superpixels with regular size. The proposed method is faster than DBSCAN [103] and produces fewer regular superpixels in textured regions, even with weak edges, achieving a more accurate delineation.

**A.3.6 F-DBSCAN.** The proposal [74] surpasses many drawbacks of the previous Real-Time DB-SCAN (RT-DBSCAN) [44] and parallelization issues. Instead of limiting the search range, the F-DBSCAN defines a limited number of points to assign for each superpixel, which minimizes the overlap and enables parallelization. The performance also maximizes the memory hints with large memory buffers, eliminating fragmentation. After the clustering step, the F-DBSCAN merges small clusters using a watershed transformation [16]. The proposal’s segmentation presents similar qualitative results to RT-DBSCAN with much faster computation. The processing time for F-DBSCAN drops as the degree of parallelism increases without increasing leakage. However, F-DBSCAN presents a poor performance in images with blue-white boundaries and low contrast due to the CIELAB colorspace used. Also, F-DBSCAN presents much slower results in GPU due to its regional parallelization instead of parallelizing a whole image.

**A.3.7 DRW.** The DRW [57] model uses dynamic nodes, which reduces the redundant calculation by limiting the walking range. The proposed algorithm performs a new seed initialization strategy that creates a seed set with regular distribution in both 2D and 3D. It also can combine boundary prior information, such as gradient information or boundary probability. [81]. DRW computes superpixels in linear time and allows control of the distribution of superpixels in complex and homogenous image regions. The proposed segmentation method has competitive performance and it is faster than existing RW models. However, DRW segmentation does not produce compact superpixels.

#### A.4 Path-based clustering

Path-based approaches generate superpixels by creating paths in the image graph based on some criteria. Usually, their clustering criteria are a path-based function to optimize during clustering. The ISF [116] is an example of a path-based method that calculates a forest of optimal paths based on a path cost function.

**A.4.1 ERGC.** First, the proposed ERGC [20] simplifies Computed Tomography (CT) images by computing superpixels based on the Eikonal algorithm. The superpixels start from seeds sampled in a regular grid and evolve according to the Fast Marching algorithm [101]. ERGC creates homogeneous superpixels with a spatial constraint to enforce compactness. The proposal allows control over the number of superpixels and compactness and is extensible to supervoxels.

**A.4.2 ISF.** Based on IFT [34], the ISF [116] framework combines a seed sampling strategy, a connectivity function, an adjacency relation, and a seed recomputation procedure. The proposal’s algorithm starts with (i) a seed sampling, followed by (ii) a spanning forest computed by the IFT algorithm, and (iii) a seed recomputation procedure. The ISF refines the segmentation by iteratively executing steps (ii) and (iii). The computational complexity of the ISF framework using a binary heap is linearithmic, independent of the number of superpixels. Also, the *Differential Image Foresting Transform* (DIFT) [26, 27, 33] can reduce the computational cost to compute the IFTs, although its effectiveness depends on the cost function used. In [116], the authors combine different components

to present five ISF-based methods. They also demonstrated that ISF produces effective and efficient methods independent of the dataset.

**A.4.3 RSS.** The RSS [21] follows the IFT [34] algorithm and can form a forest with optimal costs. To measure color similarity and spatial closeness, the authors proposed two path-based cost functions, which are more robust than the geodesic distance. Inspired by counting sort and bucket sort, the RSS computes optimal forest with buckets of queues and groups of seeds in an IFT [34]-based algorithm. Due to the sorting strategy, the proposal has  $O(n)$  complexity. The proposal is fast and has competitive performance. The main strengths of RSS are the low computational complexity, great boundary adherence with stable performance, and adjustable compactness. However, besides the proposal extends to supervoxel segmentation, it performs poorly compared with the evaluated methods. Also, due to the initial seed sampling in a regular grid [1], RSS generates more superpixels in homogenous regions, which leads to a degrading in boundary adherence in complex regions.

**A.4.4 DISF.** Based on ISF [116], DISF [14] is a three-step superpixel framework that improves its delineation even for fewer superpixels. The proposal initializes with a grid oversampling [1]. Then, iteratively compute a forest rooted at the seeds with an IFT [34] execution followed by a seed set reduction by choosing the most relevant seeds. It repeats IFT computation and seed set reduction until having the desired number of superpixels. DISF has an optimal delineation, especially for a few numbers of superpixels. Therefore, the proposal's segmentation is able to correctly select relevant seeds, reducing its boundary adherence degradation when decreasing the number of final superpixels. Despite its iterative process increasing the running time, DISF performs a reduced and limited number of iterations. However, the proposal does not produce compact superpixels.

**A.4.5 ODISF.** Motivated by OISF [12] performance, ODISF [15] extends DISF [14] for an object-based proposal to improve the superpixel performance using object saliency maps created using a U2-net [92]. The proposal performs the same three-step pipeline in DISF. First, the ODISF performs a seed oversampling. Then, it iteratively computes a spanning forest rooted at the seed set with an IFT [34] execution followed by an object-based seed removal. In the remotion step, the algorithm maintains seeds closer to the object saliency boundaries or with higher saliency. The proposed method demonstrates a generalization ability by performing an effective superpixel segmentation in datasets with different object properties. Also, it demonstrates robustness to saliency map errors in comparison with OISF. Despite the ODISF delineation step being saliency-independent, its object-based removal strategy can circumvent the saliency errors. On the other hand, the ODISF does not allow controlling the number of iterations. Also, despite its computational complexity, it has a high running time.

**A.4.6 SICLE.** SICLE [13] generalizes ODISF [15] to control the number of iterations and to improve efficiency and delineation for poorly estimated saliency maps. SICLE starts with (i) seed oversampling and iteratively generates superpixels by (ii) computing the minimum forest rooted at the seed set [34], followed by (iii) removal of the less relevant seeds. Similar to ODISF, SICLE incorporates saliency information during the seed removal step, but it is robust to incorrect saliency estimations. However, SICLE's seed removal strategy allows controlling the number of iterations and avoids unnecessary iterations, improving efficiency. Since SICLE uses object information only on the removal step, its delineation is robust to saliency errors. However, SICLE cannot improve delineation performance for more accurate saliency estimators. The authors overcome this drawback in [11] by encompassing a path cost function and a seed removal strategy to control the impact of object saliency information using a binary parameter. The proposal maintains its robustness for low-quality estimators and exploits the accurate information of high-quality estimators, improving

performance with only two iterations. Despite the robustness and efficiency of SICLE, errors in the saliency map can still affect its results.

### A.5 Hierarchical clustering

Hierarchical segmentation methods are generally not mentioned in the literature as superpixel methods. However, they fit most definitions for superpixels. Although hierarchical methods do not obtain a compact or regular segmentation, the regions produced are generally homogeneous. Furthermore, the hierarchy enables control of the desired number of regions without increasing the execution time.

**A.5.1 SH.** SH [127] uses the Boruvka algorithm to efficiently compute a *Minimum Spanning Tree* in a bottom-up manner representing a hierarchy. It improves efficiency with edge contraction, contracting each tree to a vertex and recording the edge selection order. Also, to improve accuracy with local searching, SH incorporates edge information from an edge detector and combines it with color information. In experiments, SH achieved high accuracy and low computational time. The authors also demonstrate the SH's effectiveness in saliency detection, semantic segmentation, and stereo-matching. However, SH does not produce regular superpixels.

**A.5.2 HMLI-SLIC.** HMLI-SLIC [30] consists of an (i) initial segmentation, a (ii) hierarchical multi-level segmentation, and a (iii) superpixel merging. In (i), HMLI-SLIC produces a controlled number of superpixels with SLIC [1] segmentation. Then, it performs coarse to fine segmentation to ensure that each superpixel does not contain multiple object regions, producing a hierarchical segmentation. Finally, HMLI-SLIC performs a merging step with the most similar superpixels. The proposal is robust to noise and can fit image boundaries since it produces more superpixels in heterogeneous regions and less in homogenous ones. Also, HMLI-SLIC does not perform under- or over-segmentation, automatically setting the number of seeds and superpixels. Therefore, it does not allow controlling the number of superpixels. However, the proposal is time-consuming and does not produce regular or compact superpixels.

**A.5.3 RISF.** RISF [37, 38] produces a sparse hierarchy by computing a multi-scale superpixel segmentation using ISF [116] over the Region Adjacency Graph (RAG) resulting from the previous scale. The region merging algorithm produces a dense hierarchy from a mid-level superpixel segmentation for more accurate segmentation in coarser scales. RISF produces more irregular superpixels than ISF, although it can produce a hierarchy from any superpixel segmentation method. It is also efficient, with a low complexity of  $O(n \log n)$  and its computation over RAGs. However, due to the hierarchy construction, errors in coarser scales are propagated to the finer ones.

**A.5.4 UOIFT.** UOIFT [9] extends [10] to propose a hierarchical and unsupervised image segmentation method that exploits non-monotonic-incremental cost functions in directed graphs to incorporate high-level priors of the objects as boundary polarity. UOIFT computes an initial forest over the image pixels and partitions the graph with multiple executions of the OIFT [79, 85] computed over the Region Adjacency Graph of the previous forest. UOIFT is fast and demonstrates its ability to accurately segment medical images and colored images with different lighting conditions. Although its boundary polarity allows for improving the segmentation for a specific color (or texture or local contrast) transition, setting this parameter can be challenging for more generic applications.

**A.5.5 DAL-HERS.** DAL-HERS [89] is a two-stage superpixel framework that consists of a *Deep Affinity Learning* (DAL) neural network architecture and a *Hierarchical Entropy Rate Segmentation* (HERS) method. The DAL network aggregates multi-scale information to learn pairwise pixel

affinities, and the HERS method builds a hierarchical tree structure by maximizing the graph's entropy rate. Using the DAL's affinity map, the proposed HERS algorithm constructs a hierarchy with Boruvka's algorithm [127]. The proposal preserves fine details on the objects by focusing on rich-structure parts rather than uniform regions, producing large superpixels in color-homogeneous regions and an over-segmentation in texture-rich regions. Also, compared with deep-based learning methods, DAL-HERS has a competitive running time is competitive and requires the same  $O(n)$  time complexity to produce any number of superpixels. Due to the highly adaptive nature of the produced superpixels, delineating finer details, their superpixels have no compactness.

## A.6 Density-based clustering

In the density-based clustering approach, the superpixel methods rely on an optimization function to find the cluster centers, calling them density peaks. The clustering of the non-peak pixels is performed according to the centers, generally assigning a pixel to the superpixels with the spatially closest density peak. Therefore, such methods model the problem of finding superpixels into one of finding density peaks.

**A.6.1 PGDPC.** The proposal [47] performs a two-step strategy, firstly dividing data points into peaks and non-peaks and computing a graph using DPC-based [117] allocation. Then, it classifies the peak candidates and non-peaks by computing the KNN density [132] for each pixel. After, PGDPC computes a graph based on a DPC allocation and initializes a peak graph with peak candidates as roots. The non-peak nodes are assigned to the closest root cluster, forming trees. Finally, PGDPC selects the cluster centers as candidate peaks with higher density and geodesic distances. The proposal is computationally efficient, having an  $O(n \log n)$  time complexity. In synthetic datasets, PGDPC demonstrates its ability to cluster complex structures, achieving an improved performance compared with DPC. To evaluate the proposal, the authors combined PGDPC and SLIC [1] to reduce the computational overhead. PGDPC achieves great performance in natural and medical datasets. However, using SLIC as pre-processing, the SLIC errors can be propagated to PGDPC, reducing its performance.

**A.6.2 DPS.** DPS [102] aims to perform an efficient non-iterative density peak segmentation in a limited search region. The DPS initializes computing the pixels' density and finds the density of peaks, searching in a limited region. Then, it found superpixel centers based on the pixel density and the peak density thresholds. Finally, it assigns the remaining pixels to their nearest superpixel with a higher density. Due to the regional search, its time complexity is  $O(m^2)$ , where  $m \times m$  is the region size. The proposed DPS is faster than Density Peak [95] and has a competitive segmentation, even using only color and spatial distances in a single iteration. However, DPS does not produce regular and compact superpixels. Also, its control over the number of superpixels is indirect and based on two parameters.

## A.7 Sparse linear system clustering

Sparse linear system clustering methods model the segmentation problem with a sparse matrix and extract features from linear relationships in the matrix.

**A.7.1 ANRW.** Based on the Non-local random walk (NRW) [137], the ANRW [118] performs an initial seed sampling based on the regional minima with a trade-off between local contrast and spatial distance. ANRW computes the weight matrix from the seed set according to an adaptative Gaussian function and the KNN features. It computes a Laplacian matrix and solves the Dirichlet problem, assigning labels according to it. Finally, the proposal performs a coarse-to-fine merging strategy. ANRW can deal with textured images, outperforming the compared methods in boundary

recall, under-segmentation error, and accuracy, but it has high computational complexity. Although the ANRW doesn't produce compact superpixels in complex regions, it does in homogeneous ones.

**A.7.2  $GLl_{1/2}RSC$ .** The authors [36] propose an algorithm based on subspace clustering with enhanced segmentation capability using Laplacian and  $l_{1/2}$  regularization techniques. The  $GLl_{1/2}RSC$  starts with an initial superpixel segmentation [75] and computes its Local Spectral Histogram (LSH) features to obtain a feature data matrix. Then, perform a spectral clustering on the matrix to obtain clustered data points and execute an encoding procedure to map superpixels into optimal regions [124, 148]. The proposal addresses the challenge of obtaining an improved sparse solution or a sparse representation matrix under the circumstances of noise-corrupted feature data vectors.  $GLl_{1/2}RSC$  preserves the image structures, producing better results for images with a large number of small dominant regions. However, similar to other sparse linear system clustering methods, the proposal has a high running time due to the LSH feature vector generation.

**A.7.3  $SCSC$ .**  $SCSC$  [65] formulates the superpixels problem as a subspace clustering problem. The proposed method first performs a K-means clustering. Then, it constructs a coding matrix using the superpixel-based feature vectors and solves the matrix with an algorithm based on the alternating direction method of multipliers (ADMM) [19]. Finally,  $SCSC$  computes the affinity graph and performs an NCut segmentation [105] with a further merging step to guarantee connectivity.  $SCSC$  is able to capture finer boundary details but with poor regularity and compactness. Also, it may require many seconds to generate hundreds of superpixels.

## A.8 Regional feature extraction, Polygonal decomposition, and Graph-based clustering

Regional feature extraction clustering methods iteratively extract features from image regions and use these features to perform clustering. In contrast, methods that perform Polygonal decomposition clustering decompose the image into non-overlapping polygons as superpixels. Finally, graph-based clustering methods perform superpixel segmentation based on graph topology.

**A.8.1  $EAM$ .** The proposal [4] first removes noise with a bilateral filtering [111]. Then, it extracts regional attributes using power-windows to determine whether it contain a single object. The power-windows with more than one object are iteratively split into four until achieving a minimum size. Next,  $EAM$  computes a Dijkstra [31] algorithm to merge similar power-windows, followed by a binary search to merge them with unreached windows. Finally, the proposal uses the cluster diameter threshold to control the degree of detail of segmentation.  $EAM$  is relatively fast, generates larger and fewer superpixels in homogenous regions, and captures more details in complex ones. However, the  $EAM$ 's superpixels were neither compact nor regular.

**A.8.2  $ECCPD$ .** The proposal [76] formulates the superpixel problem into a Centroidal Power Diagram (CPD) [5] problem.  $ECCPD$  starts creating fixed cluster centers for CPD using a boundary probability map from Richer Convolutional Features [72] and random centers equally spaced. Then, iteratively adapt the power cell sizes and update the power cell centers according to their centroid. After performing the maximum number of iterations or achieving the threshold,  $ECCPD$  performs post-processing to align some boundaries. Compared with other polygonal superpixel methods, the  $ECCPD$  can capture better boundaries in more complex regions. Also, the proposal is faster than other strategies to compute the CPD with capacity constraints in geometry but is highly time-consuming.

**A.8.3  $ERS$ .**  $ERS$  [4] is a greedy algorithm that efficiently computes the entropy rate of a random walk on the image graph. The  $ERS$ 's objective function is composed of an entropy rate term and a balancing term of the cluster distribution. While the entropy rate favors compact and homogeneous

clusters, the balancing term encourages clusters with similar sizes. The authors demonstrated that the balancing term in ERS produces superpixels with similar sizes and enforces control over the number of superpixels. However, they are irregular in shape.

### A.9 Data distribution-based clustering

In superpixel segmentation, we name data distribution-based methods the approaches that assume that the image pixels follow a specific distribution. From this initial conjecture, the clustering step is performed. As far as we know, the distribution-based methods that perform superpixel segmentation are based on the Gaussian mixture model and assume that the image pixels follow a Gaussian distribution.

**A.9.1 GMMSP.** GMMSP [6] models superpixel segmentation as a weighted sum of Gaussian functions, each one corresponding to a superpixel. The proposal produces superpixels of similar size by using a constant weight for the weighted sum of Gaussians. It also imposes two parameters during the expectation-maximization iterations to prevent singular covariance matrices and control superpixel regularity. GMMSP has a reduced computational complexity by using only a subset of pixels to estimate the parameters of a Gaussian function. The proposal has well-balanced accuracy and regularity but does not allow direct control over the number of superpixels. Also, GMMSP may produce irregular superpixels on strong gradient regions.

**A.9.2 gGMMSP.** To explore the parallelism in GMMSP [6], a real-time solution without the loss of segmentation consistency is proposed in [7]. The proposed gGMMSP is implemented on CUDA for GPU processing and gives very similar segmentation results as GMMSP with much faster computation. The proposal maintains the core of the GMMSP algorithm, adapting its data structures and arithmetic computations to perform GPU processing. gGMMSP requires post-processing to ensure connectivity. However, this step has data dependencies preventing parallel computing, reducing the proposal of the speedup. Even with post-processing, the gGMMSP is faster than the serial and openMP versions of GMMSP, achieving speedups of 92.6 and 27.5, respectively.

### A.10 CNN-based methods

In CNN-based superpixel segmentation, different strategies try to circumvent the limitations imposed by the rigid structure of the convolutional layers. First, we introduce SSN [55], E2E-SIS [120], BP-net [141], and DAFnet [130], which use a differential clustering module based on SLIC [1] for a pixel-superpixel assignment. Then, we discuss SEN [39] and LNSNet [147], which perform unsupervised superpixel segmentation with differential clustering modules. After, we present ML-SGN [70], SSFCN [134], SENSS [119], and AINET [126], which are u-shaped architectures. Next, we introduce ss-RIM [110], EW-RIM [136], and ML-RIM [32], which integrate the soft pixel-superpixel assignment into the convolutional process. Finally, we present SIN [138], which employs an interpolation network to enforce spatial connectivity.

**A.10.1 SSN.** The Superpixel Sampling Network (SSN) [55] is the first deep-based approach for superpixel segmentation with an end-to-end trainable pipeline that provides superpixels instead of only extracting features. In [55], the authors stated that previous superpixel algorithms are non-differentiable, making their backpropagation unfeasible. They overcome this by proposing a differentiable version of SLIC [1], where instead of a hard pixel-superpixel association, it provides a soft one. SSN has a fully convolutional network for feature extraction with seven convolutional layers interleaved with batch normalization and ReLU activations. After the second and fourth layers, a max pooling downsamples the input by a factor of two to increase the receptive field, and the input of the fourth and the sixth layers are upsampled and concatenated with the second layer's

output. The final layer output is concatenated with the XYLab of the given image and passed onto the differentiable SLIC that iteratively computes pixel-superpixel soft associations and superpixel centers. The authors demonstrated the proposal's effectiveness for specific tasks. However, the SSN does not produce connected superpixels, making it necessary to make a non-differentiable post-processing. The number of superpixels is also based on the input image dimensions, thereby not providing the exact number of desired superpixels.

**A.10.2 E2E-SIS.** The proposal [120] uses an end-to-end trainable CNN that learns deep features with two final layers, one for superpixels and the other for image segmentation. For superpixel segmentation, the deep features from the final CNN layer fed a differentiable clustering algorithm module [55]. The superpixel results and the deep features from the penultimate CNN layer are used by a superpixel pooling [62] to learn semantic similarities. The final segmentation is achieved by merging superpixels with high similarity. The E2E-SIS has a high ability to perform image and superpixel segmentation with competitive results. Since the proposal is end-to-end trainable, it can be integrated into other deep learning-based methods.

**A.10.3 BP-net.** BP-net [141] is a superpixel method for RGB-D images composed of a boundary detection network (B-net) and pixel labeling network (P-net). While the B-net learns boundaries in different scales to detect the geometry edges for depth information, the P-net extracts k-dimensional features from color information. The features extracted from P-net incorporate the geometry edge information from B-net by using a proposed boundary pass filter. The final feature map feeds a differentiable SLIC [55] to produce the final segmentation with a merging procedure, enforcing connectivity. The BP-net generates visually regular superpixels, achieving a generally reasonable regularity and capturing structured-rich regions.

**A.10.4 DAFnet.** To exploit stereo images, DAFnet [130] integrates mutual information from both image views. The proposal first extracts deep features from both image views with a weight-shared convolution network. Then, the features are integrated with a Stereo Fusion Module (SFM), composed of a Parallax Attention Module (PAM) and a Stereo Channel Attention Module (SCAM). The PAM module models the relationship between the stereo image pair to capture its spatial level correspondence, generating an attention map through a parallax-attention mechanism [121]. On the other hand, the SCAM module adaptively enhances the important information's channel [53]. Finally, inspired by SSN [55], a soft clustering module uses deep features and pixel-level information to generate the superpixels. DAFnet is the first superpixel segmentation method that extracts deep features from stereo image pairs, and its proposed PAM and SCAM modules are demonstrated to improve the results. However, it does not produce compact superpixels.

**A.10.5 SEN.** SEN [39] is an unsupervised method that learns deep embeddings using a U-net [96] architecture with a differentiable Mean-Shift clustering based on [59] for density estimation. The differentiable clustering module considers the global context, preventing embeddings from being labeled to optimize local distances. The proposal is end-to-end trainable and uses superpixel segmentation maps generated with SNIC [2] as a pseudo-ground-truth label to learn a new manifold whose feature distances act as a proxy for semantic similarity. However, SEN produces more superpixels in homogenous image regions, missing some image boundaries in complex regions.

**A.10.6 LNSNet.** LNSNet [147] is an unsupervised CNN-based method that learns superpixels in a lifelong manner. It is composed of three major modules: a feature embedder module (FEM), a gradient rescaling module (GRM), and a non-iterative clustering module (NCM). FEM embeds the original feature into a cluster-friendly space. The NCM uses the embedded features to estimate the optimal cluster centers and assigns pixel labels based on similarity. Finally, the GRM solves the

forgetting caused by lifelong learning during the backpropagation step using a Gradient Adaptive Layer (GAL) and a Gradient Bi-direction Layer (GBL). LNSNet demonstrates a high generalization capacity and generates competitive superpixels using less complex and computationally faster architecture. However, the proposal has some drawbacks. First, due to the sequential training strategy, LNSNet cannot reach a complete convergence, requiring post-processing to remove trivial regions. Also, GBL's boundary map may contain noises and lead to irregular superpixels when facing a background with a complex texture. Finally, the clustering step requires a distance matrix, which is inefficient when calculated by a CPU with a higher number of superpixels.

**A.10.7 ML-SGN.** The authors in [70] propose a multitask learning method for superpixel segmentation in SAR images. Along with superpixel segmentation, the proposed multitask learning-based superpixel generation network (ML-SGN) performs image segmentation as an auxiliary task to overcome the lack of labeled data. Inspired by [55], the ML-SGN uses a U-shaped architecture to extract features and a differential clustering strategy to produce a soft pixel-superpixel assignment. The authors employ a new distance metric based on the high-level feature space and propose a clustering module that considers high-dimensional features based on deep semantic, intensity, and spatial information. Its high-dimensional features can capture important image information, which results in highly adherent superpixels even in low-quality images. The ML-SGN is end-to-end trainable and can produce highly compact superpixels. However, it cannot directly produce superpixels.

**A.10.8 SSFCN.** The proposal [134] uses a standard encoder-decoder design with skip connections to predict association scores between pixels and regular grid cells and replace the hard pixel-superpixel assignment with a soft association map. In [134], the authors proposed two loss functions: one, similar to SLIC [1], uses an  $L_2$  norm as feature distance, and the other follows SSN [55]. Using the predicted pixel-superpixel association, SSFCN computes superpixels by assigning each pixel to the grid cell with the highest probability. The SSFCN generates compact superpixels on homogeneous image regions. The authors also demonstrate the proposal's efficacy by modifying a network architecture for stereo matching [22] to predict simultaneously superpixels and disparities. As the main drawback, the number of superpixels is controlled based on the image size and requires a post-processing step to enforce connectivity.

**A.10.9 SENSS.** SENSS [119] incorporates Squeeze-and-Excitation (SE) modules [53] into an SSFCN architecture [134]. The SE block explicitly models the inter-dependencies between channels, improving the representation power of the network. Therefore, the proposal has an encoder-decoder architecture with an attention module at each decoder block. The encoder produces high-level feature maps, and the decoder gradually upsamples the feature maps while modeling the channel-wise relationship. For training, the proposal uses the SSFCN's differentiable loss function. The proposed network outperforms the SSFCN performance, improving its learning ability with the SE blocks and achieving competitive results. However, the SE blocks have an additional computational cost. Also, SENSS has the same drawbacks as SSFCN, with limited control of the superpixels' number, and needs post-processing to guarantee connectivity.

**A.10.10 AINET.** Most deep-based superpixel models identify pixel-pixel affinities in the grid image pattern to compute superpixels instead of directly associating pixels to superpixels. In [126], an Association Implantation (AI) module is proposed to associate each pixel with its surrounding superpixels in a grid shape. They also employ a boundary-perceiving loss based on the distance between the pixel label and its reconstructed label to improve boundary delineation. The AINET is composed of a U-net architecture [96] with skip connections based on SSFCN [134], followed by an AI module. The encoder outputs a superpixel embedding in which features are propagated with a

convolution and the decoder outputs a pixel embedding. The AI module computes a pixel-superpixel association based on both embeddings. Using a loss function similar to [134], the model improves boundary precision by incorporating a boundary-perceiving loss. The AINET produces highly boundary-adherent superpixels with no compactness in textured regions, but it produces fewer thin superpixels near the strong image boundaries. Following SSFCN [134] strategy, the association map produced by AINET is based on a fixed grid sampling. Therefore, it allows partial control over the number of superpixels with image resizing, which may reduce boundary precision for images whose original size is not a multiple of the sampling spacing.

**A.10.11 ss-RIM.** Based on the idea that low-level features are insufficient to improve segmentation with few superpixels, the authors [110] induce non-local properties into an unsupervised CNN-based method. The proposal uses the *Deep Image Prior* (DIP) [63] procedure to generate task-agnostic superpixels with a new loss function based on clustering, spatial smoothness, and reconstruction. The clustering term is similar to the mutual information term of RIM [60], and the spatial smoothness cost is the same as proposed in [42]. Finally, the reconstruction cost helps the loss function fit the superpixels at the components' boundaries. The proposed method is able to generate superpixels more attached to the image boundaries, especially in heterogeneous regions. However, the ss-RIM only allows control of the upper bound number of superpixels and does not ensure connectivity.

**A.10.12 EW-RIM.** Based on ss-RIM [110], the proposal in [136] encompasses a loss function composed of four terms based on clustering [60], smooth [110], reconstruction [42], and edge-aware. The edge awareness accomplishes a differential approximation to the distribution of image gradients. Using RGB color and spatial information as input, the EW-RIM extracts feature information from a CNN architecture with a feature merging step to obtain the association probability maps. The proposed edge-aware term improves the boundary adherence of the proposed EW-RIM compared with ss-RIM, but its compactness is less. Also, since the proposal's segmentation generates more similar superpixels in size, it does not preserve finer details in complex regions.

**A.10.13 ML-RIM.** Based on ss-RIM [110] and EW-RIM [136], the authors in [32] proposed an unsupervised network for superpixel segmentation. Similar to EW-RIM, the proposed *Multi-Scale RIM* (ML-RIM) encompasses a loss function composed of four terms based on clustering [60], smooth [110], reconstruction, and edge-aware. Compared to other RIM-based approaches [110, 136], the reconstruction loss term in ML-RIM considers the mean squared error of the differentiable superpixel assignment learned by the network, and the edge-aware term employs the Kull-back-Leibler (KL) divergence loss to match between the edge distributions. In ML-RIM, edge-maps are computed using a laplacian kernel followed by a softmax. In ML-RIM, each convolutional layer is followed by an instance normalization and ReLU activation. The network is composed of a feature extractor with four convolutional layers, an ASPP module [24] to combine multi-scale features, and a final convolutional layer to transform the output features to the desired shape followed by a softmax function. ML-RIM has improved accuracy and boundary recall compared to ss-RIM and EW-RIM, and similar compactness to ss-RIM.

**A.10.14 SIN.** The SIN's [138] architecture utilizes multi-layer outputs to predict association scores using interpolations. The proposed architecture reduces the feature channels by half to extract multi-layer features with outputs to convolutional operations. Then, the convolutional operations transform the multi-layer features into 2-dimensional association scores. Finally, a pixel-superpixel map procedure uses multiple interpolations with the association scores to expand the pixel-superpixel association matrix while enforcing spatial connectivity. The initial pixel-superpixel map has a reduced size and initializes with regular sampling. The proposed method produces connected components without post-processing, being able to integrate them into downstream

tasks in an end-to-end way. SIN is faster than other deep learning-based superpixel methods, and it produces more compact and regular superpixels.

## B EVALUATION MEASURES

In general, the measures for superpixel evaluation can be divided into measures that evaluate: (i) superpixel delineation; (ii) its shape; or (iii) its color homogeneity. The delineation measures evaluate the overlap of the superpixel boundaries with the image object. The delineation-based evaluation is widespread in superpixel segmentation since the oversegmentation of the object and background regions is not penalized. On the other hand, the quality of the superpixels inside these regions is also not evaluated [109]. In this work, we evaluated boundary delineation using *Boundary Recall* (BR) [81] and *Undersegmentation Error* (UE) [87]. For color homogeneity assessment, we used the *Similarity between Image and Reconstruction from Superpixels* (SIRS) [8] and *Explained Variation* (EV) [86]. Finally, we assess superpixels' compactness using the *Compactness index* (CO) [98].

*Boundary Recall* (BR) [81] is a widely used measure for superpixel evaluation. It measures the fraction of ground-truth boundary pixels correctly detected, as presented in Equation 1, in which TP is the number of boundary pixels in a segmentation  $S$  that match the ground truth  $G$ , and FN is the number of boundary pixels in  $G$  that does not match with  $S$ . The boundary pixels are matched within a local neighborhood of size  $(2r + 1)^2$ , in which  $r$  is 0.0025 times the image diagonal.

$$\text{BR}(S, G) = \frac{\text{TP}(G, S)}{\text{TP}(G, S) + \text{FN}(G, S)} \quad (1)$$

Another widely used measure to assess the quality of superpixel segmentation delineation is the *Undersegmentation Error* (UE). Introduced by [64], the UE measures the adherence of the boundary pixels in a segmentation  $S$  to the ground truth  $G$  contours based on the area between  $S$  and  $G$  regions. UE has different versions [109]. The most recommended was proposed by [87] that evaluated the adherence to contours based on the minimum area of overlap between  $S$  and  $G$ , as presented in Equation 2, in which  $N$  is the number of pixels  $G$  and  $k$  is the number of regions in  $G$ .

$$\text{UE}(S, G) = \frac{1}{N} \sum_i^k \sum_{S_j \cap G_i \neq \emptyset} \min\{|S_j \cap G_i|, |S_j - G_i|\} \quad (2)$$

Shape-based evaluation metrics assess whether the superpixels have compact shapes with smooth contours and are arranged regularly – *i.e.*, in a grid. Although these properties have an inverse relationship to the delineation, an improved boundary recall does not necessarily imply better segmentation [98, 99]. Due to this, the quality of the superpixel methods has been evaluated in previous benchmarks according to the trade-off between its shape quality and delineation [108, 122].

The *Compactness index* (CO) [98] measure uses the isoperimetric quotient to measure the similarity between the shape of a superpixel and a circle, which constitutes the most compact geometric shape. The CO measure is presented in Equation 3, in which  $A(S_j)$  and  $P(S_j)$  are the superpixel area and perimeter, respectively.

$$\text{CO}(S) = \frac{1}{N} \sum_{S_j} |S_j| \frac{4\pi A(S_j)}{P(S_j)} \quad (3)$$

Although the desired properties of superpixels are not a consensus in the literature, the inner color similarity usually underlies their methods. The *Explained Variation* [86] defines homogeneity by comparing the variance of the superpixels' mean color  $\mu(S_i)$  and the variance of the pixels' color  $I(p)$  towards the image's mean color  $\mu(I)$ , resulting in a normalized measure (Equation 4). This measure is maximum when  $|S| = |I|$  or when  $I(p) = \mu(S_i)$  for all  $p \in S_i$  and for every  $S_i \in S$ .

However, EV considers the superpixels' mean color, which is insufficient for describing perceptually homogeneous textures [86].

$$EV(S) = \frac{\sum_{S_i \in S} |S_i| \|\mu(S_i) - \mu(I)\|_1^2}{\sum_{p \in I} \|I(p) - \mu(I)\|_1^2} \quad (4)$$

To overcome the mean color drawback, the *Similarity between Image and Reconstruction from Superpixels* (SIRS) [8] models the color homogeneity problem as an image reconstruction problem. The color descriptor *RGB Bucket Descriptor* (RBD) represents each superpixel as a small set of its most relevant colors. Let  $G^{S_i} \in S(S_i, 7)$  represent the set of 7 disjoint groups related to each RGB cube vertices, whose colors are  $c_l \in [0, 1]^3$ , in which  $1 \leq l \leq 7$ . Then, we populate each  $G_l^{S_i} \in G^{S_i}$  by assigning every  $p \in S_i$  to its most similar group using a mapping function  $M(p)$  (Equation 5).

$$M(p) = \operatorname{argmin}_{c_i \in V} \{\|x - c_i\|_1\} \quad (5)$$

The colors in RBD are used to reconstruct the original image. The reconstruction error is measured by the *Mean Exponential Error* (MEE) between the original and reconstructed image (Equation 6). The MEE increases the error weight of heterogeneous colors based on the maximum distance between the colors of the RBD. The MEE's exponent interval varies between one and two (the absolute or the mean error). Finally, SIRS defines segmentation quality as the Gaussian weighted error of reconstruction using MEE (Equation 7).

$$\text{MEE}(S) = \frac{1}{|I|} \sum_{S_i \in S} \sum_{p \in S_i} \|R(p) - I(p)\|_1^{2-\psi} \quad (6)$$

$$\text{SIRS}(S) = \exp^{-\frac{\text{MEE}(S)}{\sigma^2}} \quad (7)$$

## C ADDITIONAL EXPERIMENTAL RESULTS

This Section presents additional experimental results. First, we assess the method's ability to control the number of superpixels and to maintain connectivity. Based on the connectivity results, we perform a post-processing step to ensure connectivity for the experiments in Section 4.1 and Appendix C.2. In Appendix C.2, we evaluate the stability of superpixel methods considering the minimum, maximum, and standard deviation of Boundary Recall (BR), Undersegmentation Error (UE), Explained Variation (EV), and Similarity between Image and Reconstruction from Superpixels (SIRS). We consider a stable segmentation to have a monotonically increasing performance in those measures according to the number of superpixels. In Appendix C.3, we evaluate the robustness of superpixel methods against salt and pepper noise and average blur. Finally, Appendix C.4 reviews the overall performance of superpixel methods concerning their clustering category.

### C.1 Number of superpixels and connectivity

All superpixel methods evaluated in this work have a parameter for the desired number of superpixels, but most generate a different number than the desired one. Although control over the number of superpixels is desirable, some works reduce this control to produce a segmentation that better suits the image content. As one may note in the middle and right columns of Figure 12, most superpixel methods generate superpixels in a number close to the desired one. However, only **DISF**, **ODISF**, **SICLE**, **SH**, and **ERS** generate precisely the desired number of superpixels. In contrast, **LNSNet** and **DRW** generate quantities farther from the desired ones. While **DRW** usually produces fewer superpixels, **LNSNet** creates thousands more in the NYUV2 dataset.

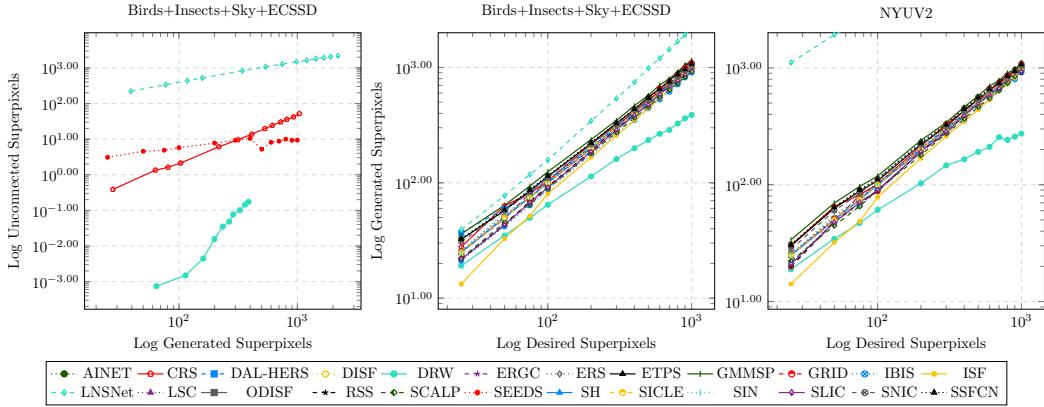


Fig. 12. Number of not connected superpixels concerning the number of generated superpixels (on the left) and the number of generated superpixels in relation to the desired number of superpixels (at the middle and right) on Birds+Insects+Sky+ECSSD and NYUV2 datasets. Note that methods without unconnected superpixels do not appear on the connectivity plot (on the left). Also, none of the methods produce unconnected superpixels in NYUV2.

Superpixel connectivity is also an important property to consider. However, many methods in the literature do not guarantee it. As one may see in the left column of Figure 12, **LNSNet**, **CRS**, **SEEDS**, and **DRW** do not guarantee the connectivity of their superpixels. While **CRS**, **SEEDS**, and **DRW** produce fewer unconnected superpixels, **LNSNet** generates much more. Only in the NYUV2 dataset, none of the methods produce unconnected superpixels. Therefore, we omit its chart. For the quantitative and stability experiments (Sections 4.1 and C.2, respectively), we perform post-processing to enforce connectivity in **LNSNet**, **SEEDS**, **DRW**, and **CRS**. Let the similarity between two superpixels as the *Euclidean distance* between their average colors. The merging step combines the smaller-area superpixels with their most similar neighbor (in an 8-neighborhood) until the number of superpixels reaches the number of segmentation labels.

## C.2 Superpixels stability

**C.2.1 Object delineation.** As one may see in Figure 13, most methods present high stability on Birds+Insects+Sky+ECSSD datasets regarding object delineation since most of them present a performance that monotonically increases in BR and decreases in UE. However, most methods are unstable in the NYUV2 dataset. In contrast, **DAL-HERS**, **SEEDS**, **ETPS**, and **CRS** show lower BR stability on all datasets. Also, **ODISF** and **SICLE** only present instability on Sky and NYUV2 datasets. The **ODISF**'s and **ODISF**'s instability explains their inferior mean BR and UE (Section 4.1) performance on those datasets. In comparison, **DAL-HERS** presents greater instability due to its creation of tiny regions, as one may see in Section 4.3. As shown in Figure 13, **DISF**, **GMMSP**, **LSC**, **SH**, and **ERS** show high stability, while **DRW** presents some instability across most datasets. **ISF** and **RSS** present stable and low std BR and UE but with some instability in max UE and min BR. **AINET** and **SSFCN** also present stability, but less than **DRW**, **ISF**, and **RSS**. A few instability was also observed on **SIN** and **SLIC**. Concerning min BR, **GRID**, **CRS**, and **SEEDS** have the worst results, while **SH**, **ISF**, **RSS**, **GMMSP**, **DISF**, **LSC**, and **ERS** have the highest ones.

**C.2.2 Color homogeneity.** Figure 14 presents the color homogeneity stability evaluation. Concerning min EV and min SIRS, most of the methods with the former have increasing values, while the

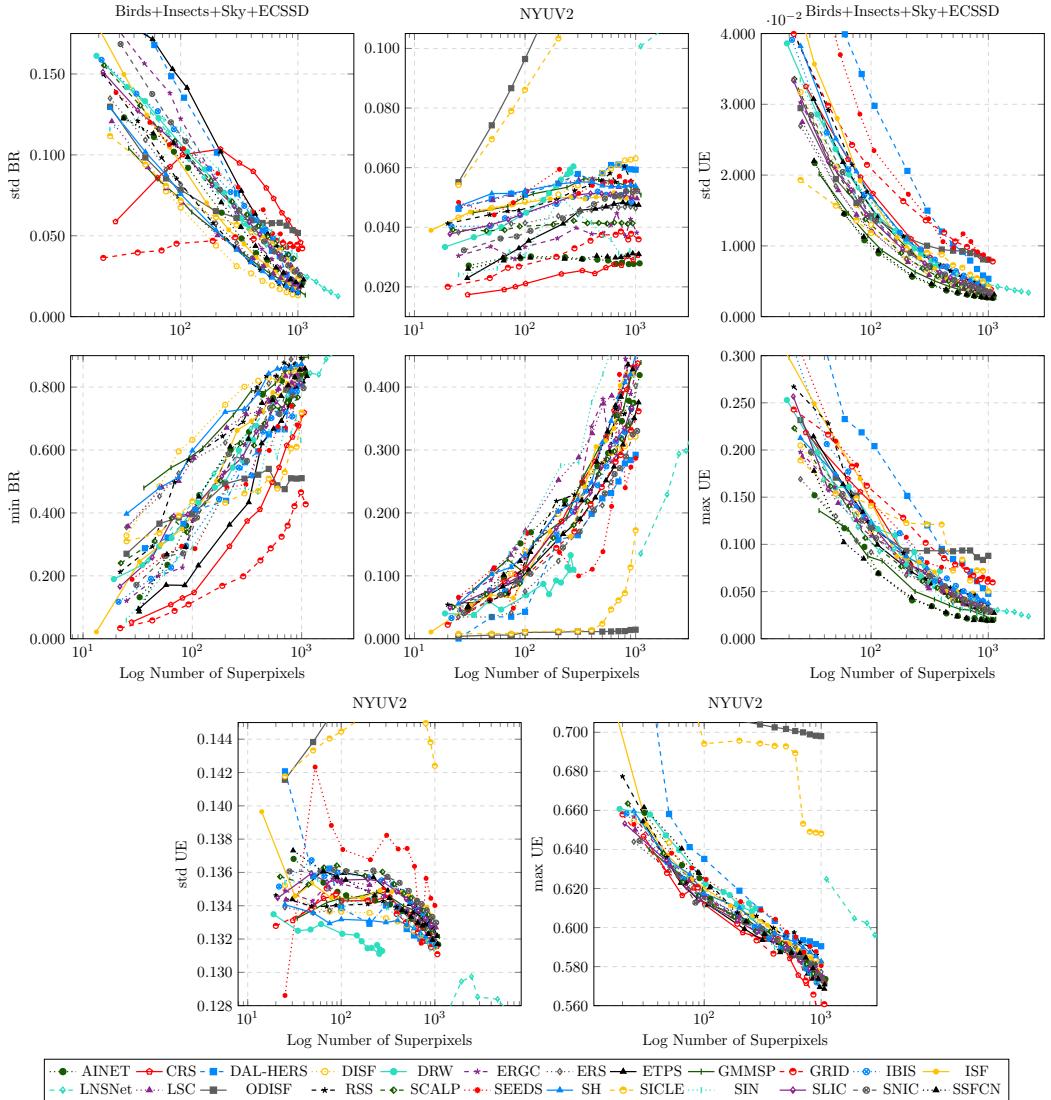


Fig. 13. Results for the minimum BR, maximum UE, and standard deviation of BR and UE on Birds+Insects+Sky+ECSSD and NYUV2 datasets.

methods with the second show more rigorous minimum scores. In both minimum measures, the methods with the highest minimum differ, except for **DISF**, which presents higher results in most datasets, followed by **SH**. Among the evaluations with min EV, **ODISF** and **SICLE** have almost constant values and worse results than **GRID** in Sky and NYUV2 datasets. These results are due to the saliency map and the concentration of superpixels in the salient region, as aforementioned. Furthermore, std SIRS and std EV also show distinct variations. While the std EV results present less stable values, the std SIRS evaluation presents more increasing results, indicating greater instability in some methods. For the std EV assessment, the methods **DISF**, **SH**, **LSC**, **SIN**, **AINET**, and **SSFCN** show high stability on all datasets. In addition, the **ISF**, **RSS**, and **SCALP** also show high stability

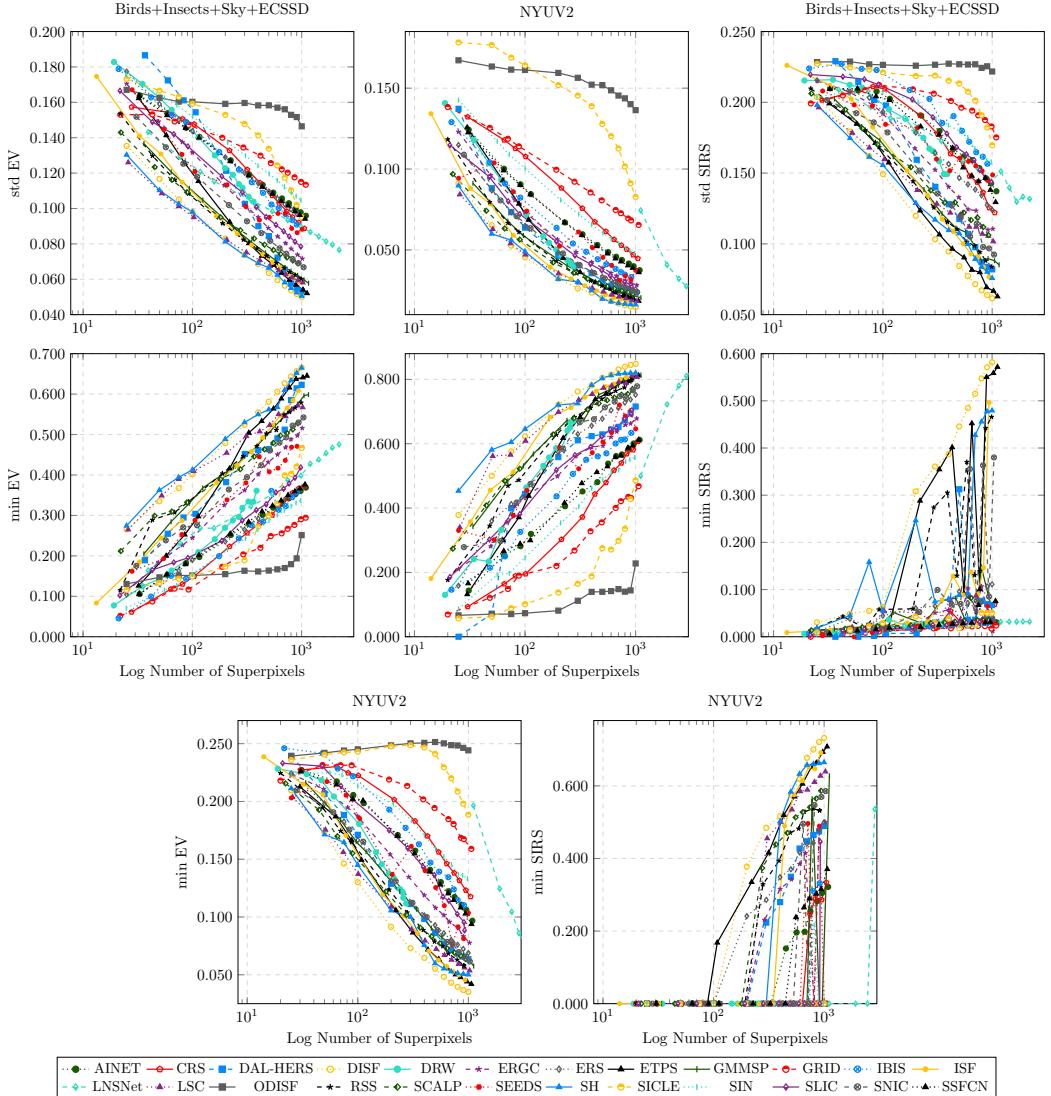


Fig. 14. Results for the minimum and standard deviation of EV and SIRS on Birds+Insects+Sky+ECSSD and NYUV2 datasets.

on at least one dataset. In std SIRS, the methods **LNSNet**, **GRID**, **IBIS**, **ODISF**, and **SLIC** show less stability on Birds and Insects datasets. On the other hand, **DISF** shows high stability in SIRS, followed by **SH** and **ETPTS**.

### C.3 Robustness

Noise and blur robustness evaluate, respectively, the susceptibility of the algorithm to strong and irrelevant edges and potentially relevant but soft edges. Similar to [109], we evaluated robustness against salt and pepper noise and average blur. In this experiment, we varied the average blur filter size by  $\{0, 5, 9, 13, 17\}$  and the noise probability by  $\{0, 0.4, 0.08, 0.12, 0.16\}$  in the Birds dataset

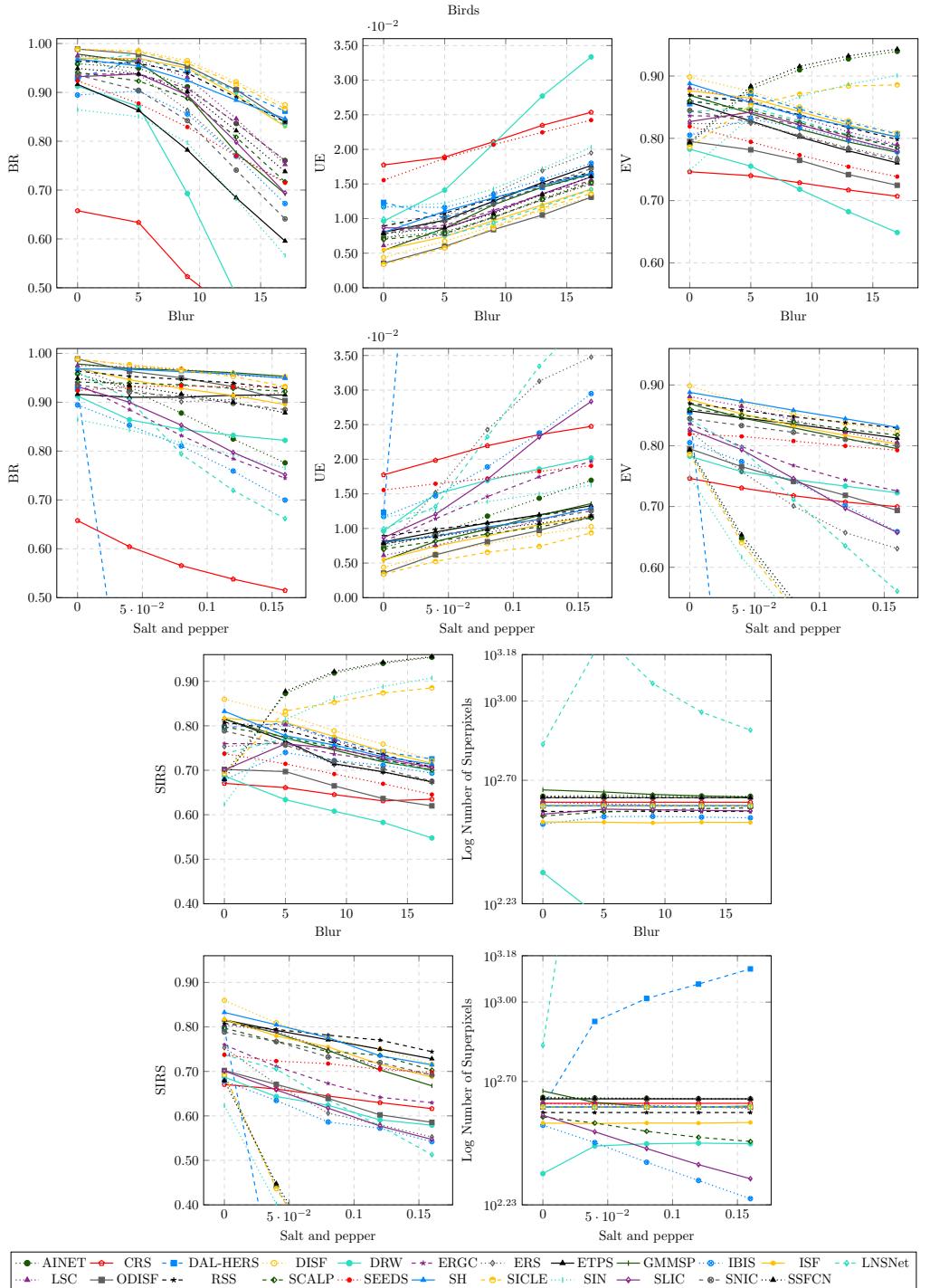


Fig. 15. Influence of average blur (first and third rows) and salt and pepper noise (second and fourth rows) for  $K \approx 400$  on Birds dataset.

images with approximately 400 superpixels. The evaluation measures used were BR, UE, EV, SIRS, and the number of superpixels produced ( $K$ ) in the segmentations.

As one may see in Figure 15, blur and noise generally tend to have a similar impact. **DISF**, **ERGC**, **RSS**, **ISF**, **ODISF**, **SEEDS**, and **SH** are robust in blur and noise. On the other hand, **DAL-HERS** shows the lowest noise robustness, followed by **LNSNet** and **ERS**. Despite being the least robust to noise, **DAL-HERS** achieves considerable robustness to blur. A similar sensitivity to noise can be observed in **SICLE**, **AINET**, and **SSFCN** regarding homogeneity. However, their homogeneity highly increased with blur. Also, they present high robustness to noise and blur concerning delineation. On the other hand, **DRW** was the most influenced by blur. One can also see that some methods present a slightly better evaluation when adding blur or noise. That is the case for **LNSNet**, **IBIS**, and **SLIC** with blur. The same occurs less perceptibly in **SEEDS**, **DAL-HERS**, **ERGC**, and **ERS**.

As shown in Figure 15, some methods try to compensate for noise and blur by producing more or fewer superpixels. Among the evaluated methods, **LNSNet** is the most impacted in the number of superpixels generated, especially when adding noise. As seen in Section C.1, **LNSNet** produces superpixels that are more discrepant in quantity, many of those disconnected. The second with the most influenced number of superpixels is **DAL-HERS** when adding noise. In addition to these, **IBIS**, **DRW**, **SLIC**, **GMMSP**, and **SCALP** show a moderate susceptibility to the number of superpixels. Finally, the addition of noise or blur does not modify the number of superpixels generated in the **CRS**, **DISF**, **ERGC**, **ERS**, **ETPS**, **SICLE**, **ODISF**, **RSS**, **SH**, and **SNIC** methods.

#### C.4 Overall performance

This Section discusses the overall performance of superpixel methods within the same clustering category. In this work, the evaluated methods with boundary evolution clustering present higher compactness, regularity, and efficiency. They nearly achieve real-time computation in ECSSD (due to its smaller images) and around 0.1 seconds per image on the others, except by **CRS** which takes around 0.3 seconds per image in ECSSD. Nevertheless, they have worse boundary adherence and color homogeneity. **IBIS** and **ETPS** have the best delineation and color homogeneity in this category. Conversely, **CRS** and **SEEDS** have the worst delineation, but the former produces the most compact superpixels. Although the superpixels in **ETPS** are not as compact as those in **CRS**, **ETPS** produces more compact superpixels than the remaining methods with boundary evolution clustering. However, **ETPS** along with **CRS** and **SEEDS** present some instability. Also, **CRS** is more sensitive to noise since its delineation results greatly decrease when increasing the average blur.

In contrast, methods with dynamic-update-clustering are less efficient and generate slightly less compact and regular superpixels. Also, they have better delineation and homogeneity than those based on boundary evolution. Although **SNIC** requires more time per image than all other CPU-based methods (around 3 to 4 seconds in most datasets compared to 1 second in **DRW**), **DRW** offers less control over the number of superpixels, being the method that produces fewer superpixels. **DRW** also produces a few unconnected superpixels, requiring post-processing. In contrast, the delineation in **DRW** usually surpasses **SNIC**, which has more compact superpixels. However, **DRW** is very sensitive to noise. By increasing average blur, **DRW** produces fewer superpixels, and they are less adherent to the objects' borders and less homogeneous.

Methods with neighborhood-based clustering present more varied performances, but similar stability. Among these, the **LSC** has more boundary adherence and homogeneity, but less compactness and smoothness than **SLIC** and **SCALP**. On the other hand, **SLIC** and **SCALP** produce more compact superpixels, the latter more than the former. Also, **SLIC** is less robust to salt and pepper noise than **LSC** and **SCALP** since its boundary adherence decreases when the noise increases.

Although **SCALP** is better at managing the tradeoff between boundary adherence and compactness, it is the less efficient method in its clustering category, requiring around one second to segment an image in most datasets, while **LSC** and **SLIC** require around 0.4 and 0.1 seconds, respectively.

The path-based clustering methods are generally stable, robust to noise, and have the best delineation along with the most homogeneous superpixels. However, they have varied efficiency, low compactness, and low smoothness. **RSS**, **ISF**, and **ERGC** produce superpixels with smooth borders. **ERGC** has worse delineation but higher compactness than **RSS** and **ISF**. Unlike **ISF**, **ERGC** does not produce highly compact superpixels at homogeneous image regions. In addition, **RSS** and **ISF** have similar good boundary adherence, high color homogeneity, and some compactness. However, **ISF** usually has better delineation, color homogeneity, and compactness than **RSS**, which is much more efficient, nearly achieving real-time processing in the **ECSSD** dataset and 0.1 seconds per image on the others. On the other hand, in most cases, **ISF** and **ERGC** take around 0.7 and 0.2 seconds, respectively. Conversely, **DISF**, **ODISF**, and **SICLE** have excellent delineation and color homogeneity in most datasets and produce the exact number of desired superpixels. However, they are less efficient than **ISF** and **RSS**, especially **ODISF** and **SICLE**. While **DISF** requires around 1.8 seconds per image, **ODISF** and **SICLE** require 2.5 seconds. Although their delineation may degrade when the saliency map fails to identify the image object, such a problem may be surpassed in **SICLE** by reducing its saliency map importance.

Hierarchical methods also produce superpixels with excellent boundary adherence and they have low execution time, but their superpixels are neither visually compact nor smooth. Among these, **DAL-HERS** has low delineation, is less stable, and is highly sensitive to salt and pepper noise, even producing much more superpixel when the noise increases. In contrast, **SH** has a competitive delineation, is stable, is more robust to noise, and is one of the most efficient methods evaluated, with nearly real-time processing. Regarding methods with clustering based on data distribution, **GMMSP** is stable, robust to noise, and has a competitive delineation. In contrast to other methods with competitive delineation, the superpixels in **GMMSP** have visually good compactness and smooth contours. However, its runtime is far from real-time, requiring around 1 second per image to produce superpixels. Similarly, **ERS**, the only evaluated method that performs graph-based clustering is stable, robust, has a competitive delineation, visually compact subpixels, and an efficiency worse than **GMMSP**, requiring around 2.5 seconds per image.

In contrast, methods that perform clustering with a deep network achieve moderate to low results. Among these, **LNSNet** presents a visually poor delineation and low compactness. It also has the worse control over the number of superpixels, since it may produce thousands more superpixels than desired. **LNSNet** has the worst efficiency and it is very sensitive to noise, generating much more superpixels when increasing average blur or salt and pepper noise. Conversely, **SSFCN** and **AINET** have similar results in all criteria. Both are sensitive to salt and pepper noise and average blur, have a moderate delineation, and have good compactness. However, they are stable and require around 0.1 seconds per image to generate superpixels. Although **AINET** slightly achieves better delineation than **SSFCN**, it is slightly less efficient. Conversely, **SIN** is a more efficient approach than the other deep-based clustering methods (except in the Birds dataset) and produces more compact superpixels, but has low boundary adherence. **SIN** is also less sensitive to salt and pepper noise than **AINET** and **SSFCN**, but is more sensitive to average blur.

Received 9 February 2023; revised 20 Febuary 2024; accepted 7 March 2024