Hierarchical Histogram Threshold Segmentation – Auto-terminating High-detail Oversegmentation

Thomas V. Chang, Simon Seibt, Bartosz von Rymon Lipinski
Game Tech Lab, Faculty of Computer Science, Nuremberg Institute of Technology, Germany
thomas.chang@th-nuernberg.de

Abstract

Superpixels play a crucial role in image processing by partitioning an image into clusters of pixels with similar visual attributes. This facilitates subsequent image processing tasks, offering computational advantages over the manipulation of individual pixels. While numerous oversegmentation techniques have emerged in recent years, many rely on predefined initialization and termination criteria. In this paper, a novel top-down superpixel segmentation algorithm called Hierarchical Histogram Threshold Segmentation (HHTS) is introduced. It eliminates the need for initialization and implements auto-termination, outperforming state-of-the-art methods w.r.t. boundary recall. This is achieved by iteratively partitioning individual pixel segments into foreground and background and applying intensity thresholding across multiple color channels. The underlying iterative process constructs a superpixel hierarchy that adapts to local detail distributions until color information exhaustion. Experimental results demonstrate the superiority of the proposed approach in terms of boundary adherence, while maintaining competitive runtime performance on the BSDS500 and NYUV2 datasets. Furthermore, an application of HHTS in refining machine learning-based semantic segmentation masks produced by the Segment Anything Foundation Model (SAM) is presented.

1. Introduction

Superpixel segmentation is an important preprocessing step in computer vision. It groups pixels with similar properties to reduce the number of primitives and enhance object representation. Various image processing tasks benefit from high-quality superpixels, such as semantic segmentation [12, 33, 42, 52], object tracking [47], object categorization [11], simultaneous localization and mapping (SLAM) [6, 20], image segmentation [10, 34, 50], video segmentation [39, 40], and stereo matching [37, 44]. There exists a wide range of approaches for image oversegmention

Figure 1. Visual comparison of 500 superpixels resulting from (a, c) ETPS [previous], (b, d) HHTS [proposed] segmentation.

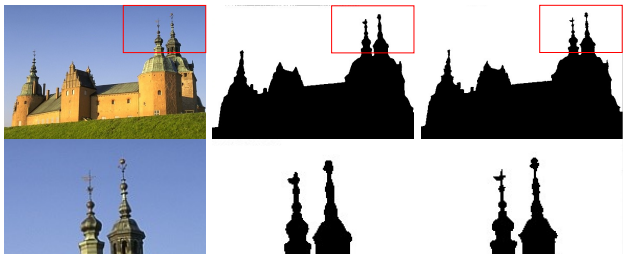

Figure 2. Visual comparison of semantic segment masks (a) original image, (b) semantic segment (SAM ViT-H) [previous] and (c) refined semantic segment (SAM + HHTS) [proposed]

[1, 5, 7–9, 18, 21, 28, 30, 46, 49, 51], often involving tradeoffs between boundary adherence, regular segment sizes, and computational efficiency. However, there are also various applications *e.g.* in computer graphics or medical imaging [37, 50] requiring the maximization of boundary recall.

In this work, a novel hierarchical oversegmentation approach based on auto-terminating local histogram thresholding is introduced, resulting in superpixels with significantly higher boundary adherence. The corresponding results can be utilized *e.g.* for fine-tuning semantic segmentation masks, as will be shown for the Segment Anything Model (SAM) [15]. Most established superpixel algorithms depend on a priori knowledge of image content and structure: Examples are seed-based methods requiring infor-

mation about the seed number and distribution [1, 5, 7–9, 17, 18, 21, 28, 28, 38, 46, 49, 51], or optimization methods relying on a pre-defined iteration limit [1, 5, 9, 17–19, 23, 28, 30, 38, 46, 49]. Proper parameterization is crucial for achieving optimal segmentation results, but often difficult to determine in practice [50]. This is true especially for seed-based methods, since an improper seed configuration may lead to detail losses [51].

The main contribution of this work comprises the Hierarchical Histogram Threshold Segmentation (HHTS) algorithm, capable of adapting to varying detail distributions and auto-terminating segmentation without prior knowledge about the input image. This is achieved through an iterative and hierarchical splitting of image regions across multiple color spaces into smaller, more homogeneous segments. It is assumed that the local histogram of each segment encodes information about local object boundaries, which are extracted iteratively, resulting in superpixels with very high boundary adherence. By utilizing multiple color spaces, various types of foreground and background can be distinguished. Larger segments, lacking recognizable boundaries, are retained in order to keep the number of image primitives low, thus potentially increasing the efficiency of segmentation applications. The only required control parameter is the smallest size in pixels of image details to capture. Additional common parameters (like the target number of superpixels) are supported, but are not mandatory. HHTS can detect color channel exhaustion and automatically stop further processing. Therewith, it can be utilized for automatic postprocessing or fine-tuning of other oversegmentation results.

Experiments demonstrate that HHTS yields superior boundary adherence results compared to recent methods, while maintaining competitive performance for multi-level segmentation. Its effectiveness is shown through the refinement of semantic segmentation masks.

2. Related Work

Image oversegmentation methods rooted in morphological image processing, such as Compact Watershed [28] and Waterpixels [21], leverage the original watershed algorithm [24] to produce fine-grained partitions of an image. These methods construct watershed lines to separate catchment basins, ultimately forming superpixels.

Graph-based superpixel techniques interpret segmentation as a graph partitioning problem. Nodes in the graph represent pixels [19], grids [49], or rectangles [38]. These nodes are merged or split based on edge weights, such as color similarity, with the aim of achieving a balanced trade-off between boundary adherence and compactness.

Clustering-based superpixel algorithms group pixels based on similarity measures in feature space, typically initiated by a seed distribution [1, 17, 18, 28, 30].

Energy optimization approaches seek an optimal labeling configuration by formulating an energy function that balances data fidelity and regularization terms. Often, the image is initially divided into regular grids to denote superpixel boundaries [9, 23, 46]. Pixels are then reassigned to neighboring superpixels iteratively based on energy.

In recent years there have been significant advances in deep learning-based superpixel techniques, resulting in approaches such as LDFUNet [8] and APENet [7]. These approaches require extensive training, but then strike a good balance between segmentation accuracy and speed [8].

There is a well-known significant issue across the aforementioned methods regarding parameter sensitivity: Each approach typically involves one or more key parameters that require careful adjustment to achieve optimal results for specific application contexts, such as medical image segmentation [50]. These parameters play a pivotal role in shaping the characteristics of the generated superpixels.

Two common aspects of parameterization are initialization and termination. Superpixel algorithms often require grids [7–9, 38, 46, 49] or seed points [1, 5, 17, 18, 28, 51], and markers [21, 28] as a starting condition. Typically, seed-based algorithms rely on complex initialization to capture details in images where information is not evenly distributed. Zhou *et al*. [51] propose a novel seed initialization solution for also capturing thin details, even for images with highly varying detail densities.

Termination is often determined by a user-defined superpixel count or maximal number of iterations [1, 5, 9, 17–19, 23, 28, 30, 38, 46, 49]. Hierarchical approaches like Superpixel Hierarchy [41] and CRTrees [45] generate multiple superpixel resolutions, providing more flexibility when a less strict termination criterion is used. Additionally, further processing steps of computer vision algorithms [13, 16, 43] can significantly benefit from such multi-scale results [41].

In this paper, a novel superpixel segmentation algorithm is presented that does not require complex initialization. It is also capable of auto-termination by analyzing locally coherent pixel color distributions, while accurately determining fine-grained segment structures.

3. Method

3.1. Multi-Channel Thresholding

The main idea of Hierarchical Histogram Threshold Segmentation (HHTS) is to extract foreground from background within a given image region by utilizing local histograms: Following the basic approach of single-color-channel histogram segmentation, an intensity threshold has to be selected that separates objects or components from their surroundings. This may work satisfactory, if objects are clearly distinguishable (*e.g*. white object on a black background in a grayscale image). To maximize the like-

lihood of finding a suitable object separation, it is proposed to employ multiple color channels and even multiple color models. Other superpixel oversegmentation approaches usually support RGB [19, 23, 46], LAB [1, 5, 9], and HSV [9, 24], each yielding different results for detecting edges and object boundaries [26]. In this work, all three color models are utilized simultaneously to capture as many object outlines as possible.

3.2. Threshold Selection

Selecting an appropriate threshold for histogram-based object segmentation is a challenging aspect, as applying simple local or global minima is usually insufficient. Traditional approaches attempt to minimize intra-class variance, such as Otsu thresholding [29] or its more recent optimization [31]. However, their ability to detect boundaries and object outlines is suboptimal (*cf*. Sec. 4).

In this work, object boundaries are identified in the histogram by interpreting intensity clusters as object classes: It is assumed that high local differences in histogram values indicate the upper or lower boundary of an object class. For boundary detection, a discrete 1D Laplace-Kernel $\begin{bmatrix}1 & -2 & 1\end{bmatrix}$ is applied to the histogram to detect thresholds candidates for object class outlines. Additionally, to prevent tiny areas being cut off from lower and upper histogram limits (*e.g*. because of blurry boundaries due to image compression), weights for a more equal partition are introduced. So, in a histogram $h = \begin{pmatrix}h_0 & h_1 & \cdots & h_{b-1}\end{pmatrix}$ with b bins, the applicability a_i – as the threshold for each bin $i \in \{0, 1, \cdots, b-1\}$ – can be determined as $a_i = l_i \cdot w_i$, where the 1D Laplace filter results l are given by: $l_i = h_{i-1} - 2 \cdot h_i + h_{i+1}$. For the equal partition weights w an adjusted Cauchy distribution is applied to the cumulative histogram $\hat{h}$ to favor thresholds near $\frac{\hat{h}_{b-1}}{2}$: $w_i = [(2 - \frac{4\hat{h}_i}{\hat{h}_{b-1}})^4 + 1]^{-1}$, with $\hat{h}_i = \sum_{j=0}^{i} h_j$. Then, the segmentation threshold t is computed as $t = \tilde{h_{i_t}}$ based on the mean histogram intensity of bin i_t with the highest threshold applicability a:

$$i_t = \arg \max_{i \in \{0,1,\cdots,b-1\}} (a_i) \tag{1}$$

The corresponding pseudocode is shown in Algorithm 2.

3.3. Hierarchical Segmentation

The segmentation of an image I with n channels $\begin{pmatrix}I_0 & I_1 & \cdots & I_{n-1}\end{pmatrix}$ is based on the following hierarchical approach: Initially, a global histogram is computed for the entire input image, applying the image thresholding procedure from Sec. 3.2. This results in two or more spatially connected segments, typically representing a separation between visually correlating components (*e.g*. similarly colored foreground objects) and the remaining image content

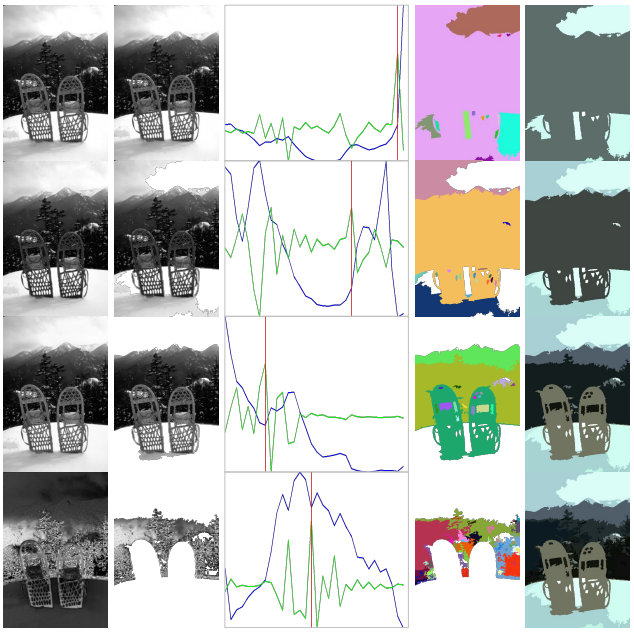

Figure 3. First four segmentation iterations with 14, 30, 48 and 92 segments: (a) most significant image channel for iteration segment, (b) iteration segment, (c) its local histogram – blue: intensity distribution, green: threshold applicability, red: separation threshold, (d) segmentation result, (e) mean segmentation image

(*e.g*. background or less distinguishable objects). This step primarily serves to delineate pixel intensity gradients in any color channel. The resulting segments encompass a smaller range of pixel intensities in at least one channel and are thus labeled as more *homogeneous*. Subsequently, the histogram threshold segmentation is performed for any resulting segment (using local segment histograms) based on a priority f_{prio}. This process iteratively builds up a segmentation hierarchy in which homogeneity increases with each successive sub-segment.

Now, let S be a set of image segments, initially comprising the entire image. For iterative superpixel segmentation, the selection of the next segment s_{next} for subdivision is determined by evaluating the priority $f_{prio}(I, s)$. It is defined by the highest per channel standard deviation $\sigma(I_j, s)$ and segment size $|s|$:

$$s_{next} = \arg \max_{s \in S} (f_{prio}(I, s)) \tag{2}$$

$$f_{prio}(I, s) = \max_{j \in \{0,1,\cdots,n-1\}} (\sigma(I_j, s)) \cdot |s|^2 \tag{3}$$

Segmentation iterations are illustrated in Fig. 3, incl. progressive extraction of image details and object outlines.

Since the entire segment selection process favors splitting smaller, more inhomogeneous segments over larger,

homogeneous ones, successive iterations automatically "fine-tune" more detailed regions while preserving larger regions with less detail. Depending on the application context, some iterations may yield too small fragments in terms of pixel area for the final segmentation result. For such cases, a user-defined threshold parameter, *detail size*, is introduced to identify too small segments, which will not be separated from their segment environment (even on detection of a potential object boundary). This approach is comparable to "forced connectivity" between small segments and their environment in SLIC postprocessing [1].

3.4. Termination Criteria

The basic iterative segmentation process continues until one of the following (combinable) termination criteria is met: *Superpixel count* – HHTS divides inhomogeneous segments until the specified number of superpixels is reached. Except for early iterations (*e.g.* < 100 segments), experiments indicate a typically accurate count of $\pm$ 2. *Segment homogeneity* – HHTS divides the most inhomogeneous segments iteratively until every segment exceeds the specified homogeneity threshold (*cf*. Eq. (3)).

3.5. Auto-terminating Segmentation

HHTS also supports "information-exhaustive" oversegmentation, which does not require parameter adjustment, nor prior knowledge of input data for termination: If during iterative execution a segment comprises more than one pixel intensity in any color channel, it is considered as (still) splittable. Otherwise, it is marked as final, since its channel information has been exhausted. Upon exhausting all splittable segments, execution terminates automatically, yielding the final oversegmentation result.

Note that after termination, every pixel cluster that is smaller than *detail size* remains connected to its adjoining neighboring pixels, preserving object coherence according to Sec. 3.2. Consequently, each final superpixel contains clusters of identical pixels (with the same intensity in every color channel) alongside clusters of corresponding fine-granular details. Refer to Algorithm 1 and 2 for the complete HHTS procedure and segment splitting, respectively.

3.6. Extension – Superpixel Refinement

HHTS can be applied to non-rectangular or (binary) masked sub-images of any shape. Thus, its auto-termination can be employed to auto-refine other oversegmentation approaches as follows: Instead of "guessing" any fine-tuned parametric adjustment relevant for initialization or termination (*e.g.* a high superpixel count), a less detailed segmentation (*i.e.* easier to estimate low superpixel count) is performed with a fast oversegmentation algorithm. Then, its results are passed to HHTS for auto-segmenting the remaining details within the initial superpixel structure.

Algorithm 1 HHTS segmentation algorithm

Input: Image I, histogram bins b, min detail size m, [termination criterion $c = false$,] [pre-segmentation $P = \{I\}$]
Hist: Histogram Calculation Algorithm
Output: Superpixel labels S

$S \leftarrow P$
$s_{next} \leftarrow \arg\max_{s \in S} f_{prio}(I, s)$
while $c(s_{next}) = false$ **do**
  $S \leftarrow S - \{s_{next}\}$
  $h_{next} \leftarrow Hist(s_{next}, I, b)$
  if $|h_{next}| < 2$ **then**
    break
  end if
  $U_{low}, U_{high} \leftarrow Split(s_{next}, h_{next}, m)$
  $S \leftarrow S \cup U_{low} \cup U_{high}$
  $s_{next} \leftarrow \arg\max_{s \in S} f_{prio}(I, s)$
end while

Algorithm 2 HHTS segment split algorithm *Split(s, h, m)*

Input: Segment s, Histogram h, min detail size m
CC: Connected Components [4]
Output: Segments below threshold U_{low}, Segments above threshold U_{high}

$i_t \leftarrow \arg\max_{i \in \{0,1,\cdots,|h|-1\}}(a_i)$
$t \leftarrow \tilde{h_{i_t}}$
$U_{low} \leftarrow CC(s < t)$
$U_{high} \leftarrow CC(s \geq t)$
$U_{low}^{small} \leftarrow \{u \in U_{low} \mid |u| < m\}$
$U_{high}^{small} \leftarrow \{u \in U_{high} \mid |u| < m\}$
while $U_{low}^{small} \neq \emptyset$ **and** $U_{high}^{small} \neq \emptyset$ **do**
  $U_{low} \leftarrow CC(U_{low} - U_{low}^{small} \cup U_{high}^{small})$
  $U_{high} \leftarrow CC(U_{high} - U_{high}^{small} \cup U_{low}^{small})$
  $U_{low}^{small} \leftarrow \{u \in U_{low} \mid |u| < m\}$
  $U_{high}^{small} \leftarrow \{u \in U_{high} \mid |u| < m\}$
end while

4. Experiments

For HHTS evaluation, the superpixel benchmark proposed by Stutz *et al.* [36] is used on the BSDS500 [2] and NYUV2 [35] datasets. BSDS500 comprises 500 outdoor nature images, while NYUV2 consists of 1449 depth images, both providing ground truth segment annotations. All experiments were conducted on a laptop with an Intel i9-13900HX at 3.8GHz, 64GB RAM, and a mobile RTX 4070.

4.1. Comparison with State-of-the-Art Methods

The oversegmentation results are compared quantitatively using common evaluation metrics, including: (1) Boundary Recall (BR) [22], (2) Undersegmentation Error (UE) [1, 17, 27], (3) Mean Distance to Edge (MDE) [3], (4) Achievable

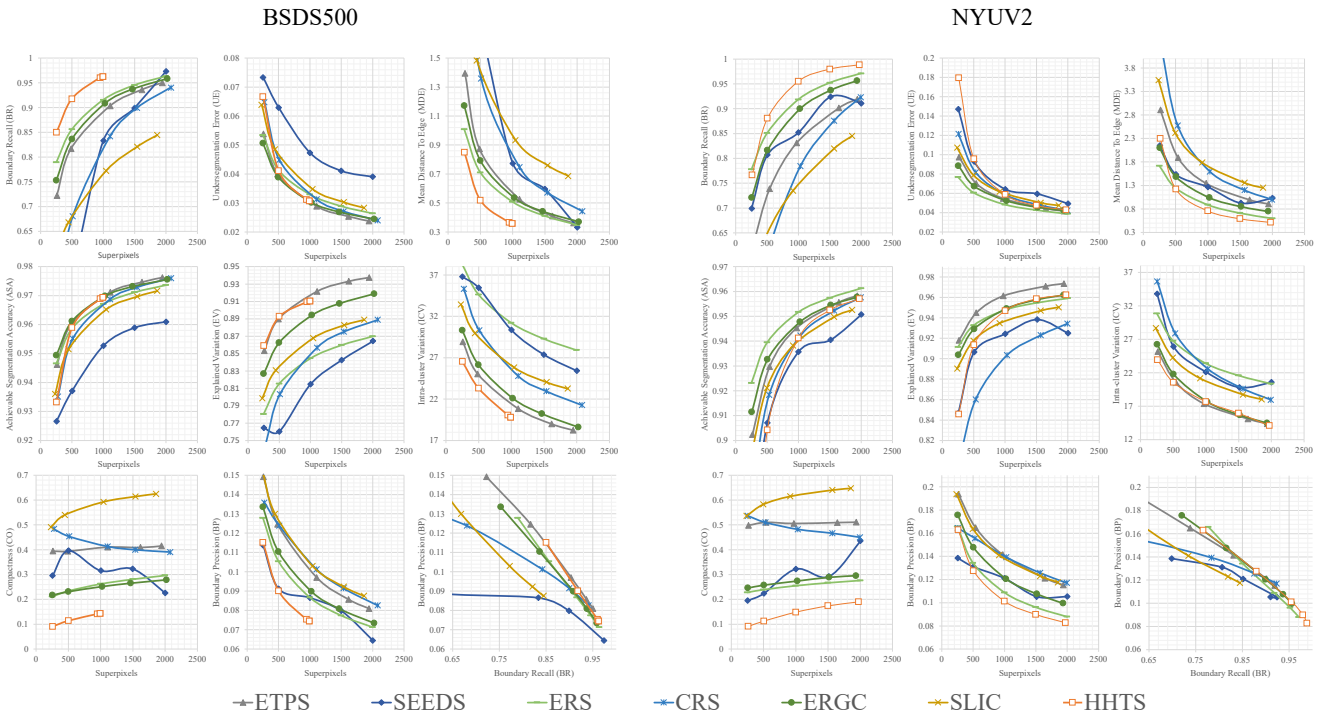

Figure 4. Benchmark results for BSDS500 and NYUV2 datasets with default parameterization [36] in comparison to proposed HHTS

Superpixels	Method	UE	BR	ASA	EV	CO	BP
250	SH [41]	0.0970	0.8080	**0.9510**			
	HHTS	**0.0668**	**0.8502**	0.9332			
600	SCAC [48]	0.0680	0.8260	**0.9660**	0.8750	**0.4420**	
	HHTS	**0.0373**	**0.9326**	0.9627	**0.8989**	0.1215	
1000	VSSS [51]	0.0324	0.9188	0.9676	**0.9123**	**0.1953**	
	HHTS	**0.0307**	**0.9626**	**0.9693**	0.9100	0.1411	
1200	APENet [7]		0.9204	**0.9758**			**0.1878**
1000*	HHTS		**0.9626**	0.9693			0.0744
1300	LDFUNet [8]		0.9300	**0.9734**			**0.0996**
1000*	HHTS		**0.9626**	0.9693			0.0744
2000	CRTREES [45]	0.0716	0.9624		**0.9482**		
1000*	HHTS	**0.0307**	**0.9626**		0.9100		

Table 1. Comparison of state-of-the-art superpixel methods to HHTS (BSDS500 dataset); "*" indicates HHTS early-termination

Segmentation Accuracy (ASA) [3], (5) Explained Variation (EV) [25], (6) Intra-Cluster Variation (ICV) [3], (7) Compactness (CO) [32], (8) Boundary Precision (BP).

HHTS is evaluated against popular algorithms, implemented in Stutz's *et al*. superpixel benchmark [36] for the test sets of BSDS500 (205 images) and NYUV2 (399 images): ETPS [46], SEEDS [9], ERS [19], CRS [23], ERGC [5] and SLIC [1]. Results from more recent state-of-the-art oversegmentation approaches, including deep learning-based methods, were compared using their respective published findings: SH [41], SCAC [48], VSSS [51], APENet [7], LDFUNet [8] and CRTrees [45]. For all measurements, HHTS is executed with default parameters unless stated otherwise: 32 histogram bins, 64 pixels minimum detail size. Note that HHTS auto-terminates processing (*cf*. Sec. 3.5) when reaching high superpixel counts (*e.g*. $800 - 1200$ segments for BSDS500 images).

4.1.1 Quantitative Result Comparison

Fig. 4 shows the HHTS superpixel performances for various segment counts in Sutz's *et al*. benchmark [36], and Tab. 1 a comparison to more recent approaches. The segmentation results obtained with HHTS reveal significantly higher boundary adherence (*cf*. BR and MDE) and lower intra-cluster variation (ICV), outperforming every other method, except ERS with less than 500 superpixels for NYUV2 and tieing with ETPS on the NYUV2 dataset, respectively. This superior results are attributed to the priority-driven and progressive refinement of segments, supporting the multi-channel histogram-based extraction of local objects and their outlines, respectively (Sec. 3). However, this achievement comes at the cost of lower compactness (CO), primarily due to the irregular shapes of HHTS superpixels: Elongated segments contribute to an increase in the relative size of boundaries, consequently decreasing boundary precision (BP). Despite this trade-off, the balance between BP and boundary recall (BR) remains superior, comparable to established methods like ETPS. Regarding the other superpixel metrics (*e.g*. UE, ASA and EV), the proposed method achieves comparable results to leading algorithms.

In general, HHTS performs better particularly for higher superpixel counts, underlying its effectiveness in capturing finer objects parts. The top-down approach of HHTS proves beneficial especially in adapting to varying distributions of details across images. This addresses the major challenge of seed-based methods, as discussed in Sec. 3.3 and mentioned in [51]. Additionally, preserving large homogeneous areas allows to reduce the overall number of superpixels without compromising the segmentation quality in terms of the aforementioned metrics. Fewer but larger superpixels can expedite subsequent image processing steps, such as region merging [43] or superpixel matching [6], despite a higher variance in superpixel sizes and a lower compactness.

On the other hand, challenges arise during early iterations of histogram threshold selection, especially if the input image contains highly distinguishable regions, that are not included in the ground truth segmentation (such as lights and bright reflections), or excessively smooth object transitions (*e.g*. due to similar image contents, low image resolution, or data compression artifacts). The latter poses a similar issue as faced by CRTrees [45], resulting in boundary leakage (since such object outlines are hard to detect using primarily color distribution information). So, early HHTS oversegmentation iterations may contain superpixels that expand over multiple ground truth segments, in particular increasing the undersegmentation error (UE). Approaches favoring uniform segment sizes and compactness (typically due to rectangular seed initialization [46, 49] or optimization regulations and constraints [46, 51]) often limit segment leakages, though sometimes at the expense of less distinct or even arbitrary superpixel borders [9].

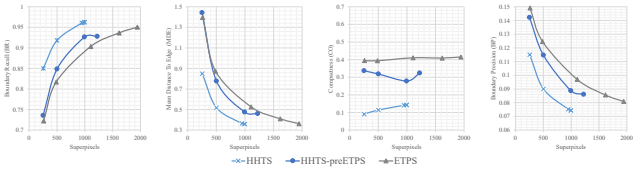

Figure 5. BSDS500 results for HHTS refinement of ETPS (1:1)

(a) (b) (c) (d)

Figure 6. HHTS auto-refinement of initial ETPS segments (a) 250 → 1312, (b) 500 → 1355, (c) 1000 → 1470 and (d) original image with 1500 ETPS segments as reference

Utilizing HHTS superpixel-refinement for pre-segmented images (*cf*. Sec. 3.6) makes it possible control the trade-off between boundary adherence and superpixel regularity: Fig. 5 shows the results of this approach exemplary for "HHTS-preETPS", *i.e*. a ETPS pre-segmentation (with 50% of superpixels) is subsequently refined with HHTS (contributing also 50% of superpixels). On the one hand, this results in a substantial increase in compactness and boundary precision w.r.t. pure HHTS, while in average being 26.3% faster for BSDS500 images. On the other hand, HHTS-preETPS achieves higher boundry recall and a lower mean distance to edge w.r.t. pure ETPS.

Based on an exemplar segmentation case with 600 superpixels, runtime measurements of HHTS yielded an average of about 448.0 ms per image for the BSDS500 dataset (single-threaded and CPU-only implementation). The resulting hierarchical segmentations comprised 172 levels of segmentation per image (between 10 and 365), so an overall average performance of 2.6 ms per image and level. For a comparable experimental setup, the other approaches achieved the following average results for a single level per image: SEEDS 14.3 ms, ETPS 19.4 ms, SLIC 30.3 ms, ERGC 57.3 ms, CRS 156.4 ms and ERS 243.3 ms. For superpixel segmentation across multiple levels of detail (*e.g*. for [16, 43]), the other approaches were capable of gener-

Figure 7. Mean color BSDS500 images at 500 superpixels: (a) ERS, ERGC, SEEDS, SLIC, CRS (two each), (b) ETPS, and (c) HHTS

Figure 8. Mean images for HHTS segmentation: (a) 200, (b) 300, (c) 400, (d) 500 superpixels, and (e) original image

ating only a fraction of the 172 levels per BSDS500 image in the given HHTS execution time (448.0 ms): SEEDS 31 levels (18.2%), ETPS 23 (13.4%), SLIC 14 (8.6%), ERGC 7 (4.5%), CRS 2 (1.7%) and ERS just one level (1.1%).

4.1.2 Qualitative Result Comparison

Fig. 1 and Fig. 7 provide a visual comparison between the proposed method and other oversegmentation algorithms for 500 superpixels. The images underline HHTS's capability of improved preservation of object boundaries. Fig. 8 illustrates the progressive improvement in visual quality achieved by HHTS through multiple iterations. Each iteration refines the oversegmentation, capturing increasingly intricate image color gradients and demonstrating the ability to extract fine details with each successive step. Fig. 6 shows the results of auto-terminating superpixel-refinement of ETPS [46]. This illustrates HHTS's visual adaptability in pre-segmentation enhancement across various superpixel resolutions, capturing even thin details.

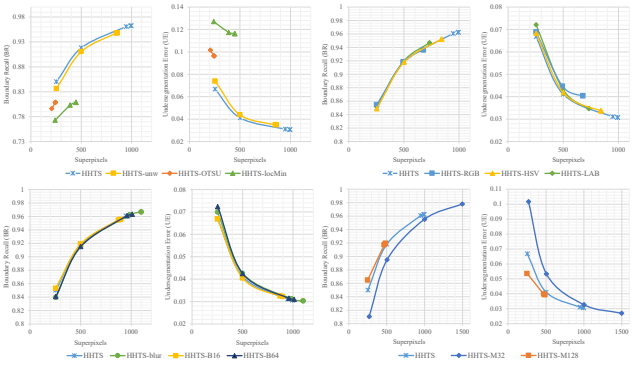

Figure 9. The ablation study of HHTS on BR and UE

4.2. Ablation Study

The following HHTS ablation study was conducted using the BSDS500 dataset, focusing on the boundary recall (BR) and undersegmentation error (UE): First, instead of the object-based histogram thresholding method described in Sec. 3.2, two alternative thresholding approaches were explored: OTSU thresholding [29] to minimize intra-class variance ("HHTS-OTSU"), and local histogram minima, descending from the histogram center ("HHTS-locMin"). Results revealed a significantly inferior oversegmentation in terms of BR and UE for both alternatives (*cf*. Fig. 9) and a tendency to early-terminate due to tiny segments (*cf*. detail size). The results underline the effectivity of the proposed thresholding approach at any superpixel scale.

By disabling equal partition weights during threshold selection (Sec. 3.2), HHTS is more likely to extract tiny segments with excessively high or low pixel intensities, respec-

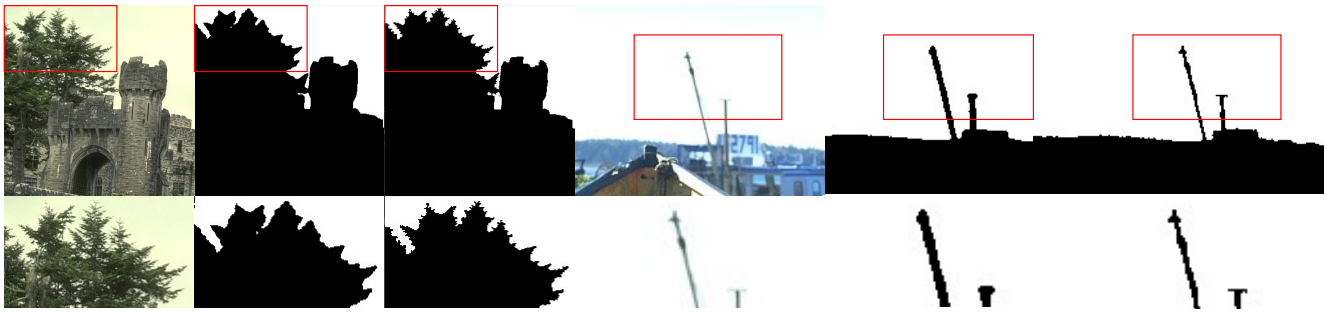

Figure 10. Visual comparison of semantic segment masks (SAM ViT-H) and refined semantic segments (SAM + HHTS)

tively. This may include leftover boundary pixel fragments from previous iterations, if the selected threshold could not separate objects and backgrounds satisfactory. Corresponding results show slightly poorer BR and UE values for disabled equal partition weights ("HHTS-unw").

To assess the impact of different image color spaces, HHTS was executed using just RGB ("HHTS-RGB"), HSV ("HHTS-HSV"), and LAB ("HHTS-LAB"). The results, also presented in Fig. 9, indicate a BR and UE comparable to the proposed approach. However, using the full multi-channel approach leads to a clearly higher total number of achievable superpixel counts. This may be explained by HHTS-RGB/-HSV/-LAB having limited options during segmentation threshold selection, potentially leading to the extraction of less relevant objects.

Applying Gaussian blurring, exemplary with a 3×3 kernel ("HHTS-blur"), or altering the number of histogram bins to 16 and 64 ("HHTS-B16" and "HHTS-64") had only a negligible impact on UE and BR, underscoring the algorithm's robustness against bad parameter choices.

The minimum detail size significantly influenced HHTS: Increasing the minimum detail size, *e.g.* to 128 pixels, improved BR and UE, but reduced the number of extracted segments. Conversely, allowing smaller details, *e.g.* 32 pixels, decreased BR and UE performance in early iterations due to the extraction of tiny objects that potentially are not included in the ground truth segments. However, this enables the detection of more detailed structures during later iteration, resulting in an overall higher BR and UE performance at the expense of a higher superpixel count.

5. Application Example

Given the high boundary recall of HHTS, an application for postprocessing semantic segmentations is proposed: The primary emphasis lies on machine-learning-based methods and particularly the large-scale Foundation Model SAM [15], since the resulting segmentation masks are prone to coarse boundaries, especially for thin components [14]. Fine-tuning of such masks is implemented by adjusting the corresponding pixel set towards the nearest oversegmentation boundaries through a combination of dilation and erosion operations. These morphological transformations are performed for mask regions intersecting superpixels and depending on whether the majority of the superpixel is inside (dilate) or outside the mask (erode). A visual evaluation of HHTS mask-refinement based on BSDS500 images and SAM ViT-H model (for 500 superpixels) resulted in a significantly improved segmentation fidelity w.r.t. fine-granular image structures (*cf*. Fig. 2 and Fig. 10).

6. Conclusion

In this paper, HHTS, a novel superpixel method based on hierarchical histogram threshold segmentation is presented. Automatic separation between foreground and background pixel regions is achieved through iterative processing of local histograms. HHTS supports auto-termination w.r.t. to a user-controllable minimum detail size, eliminating the need for seed distribution initialization or specification of secondary termination criteria, such as a maximum superpixel count. Experimental results demonstrate HHTS's superior segmentation performance in terms of boundary adherence at a competitive speed for multi-level segmentation. Moreover, applications are proposed for refining pre-segmented superpixel results and fine-tuning semantic segmentation masks based on the "Segment Anything Model" (SAM).

Future work includes optimizing the runtime of local histogram computation and connected components estimation, *e.g.* by utilizing graph-cuts. Leveraging HHTS's high boundary adherence in conjunction with its hierarchical multi-scale approach, integration with dedicated saliency segmentation methods [13, 16, 43] can further refine semantic masks, offering an additional layer of precision and adaptability. Ongoing research also comprises dynamically adjusting the minimum detail size parameter, incorporating depth information for histogram thresholding, and implementing time-based termination for interactive applications.

Acknowledgements

This work is funded by the Federal Ministry of Education and Research (BMBF Germany, No. 01IS23007B).

References

[1] Radhakrishna Achanta, Appu Shaji, Kevin Smith, Aurelien Lucchi, Pascal Fua, and Sabine Süsstrunk. Slic superpixels compared to state-of-the-art superpixel methods. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(11):2274–2282, 2012. 1, 2, 3, 4, 5

[2] Pablo Arbeláez, Michael Maire, Charless Fowlkes, and Jitendra Malik. Contour detection and hierarchical image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(5):898–916, 2011. 4

[3] Wanda Benesova and Michal Kottman. Fast superpixel segmentation using morphological processing. In *Conference on Machine Vision and Machine Learning*, 2014. 4, 5

[4] Federico Bolelli, Stefano Allegretti, and Costantino Grana. One dag to rule them all. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021. 4

[5] P. Buyssens, M. Toutain, A. Elmoataz, and O. Lezoray. Eikonal-based vertices growing and iterative seeding for efficient graph-based segmentation. In *IEEE International Conference on Image Processing*, 2014. 1, 2, 3, 5

[6] Alejo Concha and Javier Civera. Using superpixels in monocular SLAM. In *IEEE International Conference on Robotics and Automation*, 2014. 1, 6

[7] Yuanyuan Dang, Yu Gao, Bing Liu, and Chenxi Li. APENet: Adaptive pyramid pooling and SE network for superpixel segmentation. In *International Conference on Computer Vision, Image and Deep Learning*, 2023. 1, 2, 5

[8] Yuanyuan Dang, Junheng Wei, Lijie Zhang, and Zhaohao Zhong. LDFUNet: Large-kernel convolution and dynamic fusion network for superpixel segmentation. In *International Conference on Computer Vision, Image and Deep Learning*, 2023. 2, 5

[9] Michael Van den Bergh, Xavier Boix, Gemma Roig, and Luc Van Gool. SEEDS: Superpixels extracted via energy-driven sampling. *International Journal of Computer Vision*, 111(3):298–314, 2014. 1, 2, 3, 5, 6

[10] Amal Farag, Le Lu, Holger R. Roth, Jiamin Liu, Evrim Turkbey, and Ronald M. Summers. A bottom-up approach for pancreas segmentation using cascaded superpixels and (deep) image patch labeling. *IEEE Transactions on Image Processing*, pages 386–399, 2017. 1

[11] Brian Fulkerson, Andrea Vedaldi, and Stefano Soatto. Class segmentation and object localization with superpixel neighborhoods. In *IEEE International Conference on Computer Vision*, 2009. 1

[12] Raghudeep Gadde, Varun Jampani, Martin Kiefel, Daniel Kappler, and Peter V. Gehler. Superpixel convolutional networks using bilateral inceptions. In *European Conference on Computer Vision*. 2016. 1

[13] Huaizu Jiang, Jingdong Wang, Zejian Yuan, Yang Wu, Nanning Zheng, and Shipeng Li. Salient object detection: A discriminative regional feature integration approach. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2013. 2, 8

[14] Lei Ke, Mingqiao Ye, Martin Danelljan, Yifan Liu, Yu-Wing Tai, Chi-Keung Tang, and Fisher Yu. Segment anything in high quality. In *Neural Information Processing Systems*, 2023. 8

[15] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C. Berg, Wan-Yen Lo, Piotr Dollar, and Ross Girshick. Segment anything. In *IEEE International Conference on Computer Vision*, 2023. 1, 8

[16] Pushmeet Kohli, L'ubor Ladický, and Philip H. S. Torr. Robust higher order potentials for enforcing label consistency. *International Journal of Computer Vision*, 82(3):302–324, 2009. 2, 6, 8

[17] A. Levinshtein, A. Stere, K.N. Kutulakos, D.J. Fleet, S.J. Dickinson, and K. Siddiqi. TurboPixels: Fast superpixels using geometric flows. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(12):2290–2297, 2009. 2, 4

[18] Zhengqin Li and Jiansheng Chen. Superpixel segmentation using linear spectral clustering. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2015. 1, 2

[19] Ming-Yu Liu, Oncel Tuzel, Srikumar Ramalingam, and Rama Chellappa. Entropy rate superpixel segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2011. 2, 3, 5

[20] Ran Long, Christian Rauch, Tianwei Zhang, Vladimir Ivan, Tin Lun Lam, and Sethu Vijayakumar. Rgb-d slam in indoor planar environments with multiple large dynamic objects. *IEEE Robotics and Automation Letters*, 7(3):8209–8216, 2022. 1

[21] V. Machairas, E. Decenciere, and T. Walter. Waterpixels: Superpixels based on the watershed transformation. In *IEEE International Conference on Image Processing*, 2014. 1, 2

[22] D.R. Martin, C.C. Fowlkes, and J. Malik. Learning to detect natural image boundaries using local brightness, color, and texture cues. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(5):530–549, 2004. 4

[23] Rudolf Mester, Christian Conrad, and Alvaro Guevara. Multichannel segmentation using contour relaxation: Fast superpixels and temporal propagation. In *Image Analysis*, pages 250–261. 2011. 2, 3, 5

[24] Fernand Meyer. Color image segmentation. In *International Conference on Image Processing and its Applications*, pages 303–306. IET, 1992. 2, 3

[25] Alastair P. Moore, Simon J. D. Prince, Jonathan Warrell, Umar Mohammed, and Graham Jones. Superpixel lattices. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2008. 5

[26] Prachi R. Narkhede and Aniket V. Gokhale. Color image segmentation using edge detection and seeded region growing approach for CIELab and HSV color spaces. In *IEEE International Conference on Industrial Instrumentation and Control*, 2015. 3

[27] Peer Neubert and Peter Protzel. Superpixel benchmark and comparison. In *Proc. of Forum Bildverarbeitung*, pages 1–12. 4

[28] Peer Neubert and Peter Protzel. Compact watershed and preemptive SLIC: On improving trade-offs of superpixel segmentation algorithms. In *International Conference on Pattern Recognition*, 2014. 1, 2

[29] Nobuyuki Otsu. A threshold selection method from gray-level histograms. *IEEE Transactions on Systems, Man, and Cybernetics*, 9(1):62–66, 1979. 3, 7

[30] Jeremie Papon, Alexey Abramov, Markus Schoeler, and Florentin Worgotter. Voxel cloud connectivity segmentation - supervoxels for point clouds. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2013. 1, 2

[31] Khamael Raqim Raheem and Hafedh Ali Shabat. An otsu thresholding for images based on a nature-inspired optimization algorithm. *Indonesian Journal of Electrical Engineering and Computer Science*, 31(2):933, 2023. 3

[32] Alexander Schick, Mika Fischer, and Rainer Stiefelhagen. Measuring and evaluating the compactness of superpixels. In *IEEE International Conference on Pattern Recognition*, 2012. 5

[33] Abhishek Sharma, Oncel Tuzel, and Ming-Yu Liu. Recursive context propagation network for semantic scene labeling. *Advances in Neural Information Processing Systems*, 27, 2014. 1

[34] Jianbing Shen, Xingping Dong, Jianteng Peng, Xiaogang Jin, Ling Shao, and Fatih Porikli. Submodular function optimization for motion clustering and image segmentation. *IEEE Transactions on Neural Networks and Learning Systems*, 30(9):2637–2649, 2019. 1

[35] Nathan Silberman, Derek Hoiem, Pushmeet Kohli, and Rob Fergus. Indoor segmentation and support inference from RGBD images. In *European Conference on Computer Vision*, pages 746–760. 2012. 4

[36] David Stutz, Alexander Hermans, and Bastian Leibe. Superpixels: An evaluation of the state-of-the-art. *Computer Vision and Image Understanding*, 166:1–27, 2018. 4, 5, 6

[37] Longhao Sun. Multi-path minimum spanning tree and superpixel based cost aggregation for stereo matching. *IEEE Access*, 11:121096–121108, 2023. 1

[38] Olga Veksler, Yuri Boykov, and Paria Mehrani. Superpixels and supervoxels in an energy optimization framework. In *European Conference on Computer Vision*. 2010. 2

[39] Wenguan Wang, Jianbing Shen, Ruigang Yang, and Fatih Porikli. Saliency-aware video object segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(1):20–33, 2018. 1

[40] Wenguan Wang, Jianbing Shen, Fatih Porikli, and Ruigang Yang. Semi-supervised video object segmentation with super-trajectories. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41(4):985–998, 2019. 1

[41] Xing Wei, Qingxiong Yang, Yihong Gong, Narendra Ahuja, and Ming-Hsuan Yang. Superpixel hierarchy. *IEEE Transactions on Image Processing*, 27(10):4838–4849, 2018. 2, 5

[42] Wu Xiaoning, Li Ruixin, Zhu Xinli, Zhu Siyan, and Cui Daowang. Iterative weakly supervised semantic segmentation network with fused superpixel clues. In *International Conference on Natural Computation, Fuzzy Systems and Knowledge Discovery*, 2023. 1

[43] Qiong Yan, Li Xu, Jianping Shi, and Jiaya Jia. Hierarchical saliency detection. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2013. 2, 6, 8

[44] Tingman Yan, Yangzhou Gan, Zeyang Xia, and Qunfei Zhao. Segment-based disparity refinement with occlusion handling for stereo matching. *IEEE Transactions on Image Processing*, 28(8):3885–3897, 2019. 1

[45] Tingman Yan, Xiaolin Huang, and Qunfei Zhao. Hierarchical superpixel segmentation by parallel CRTrees labeling. *IEEE Transactions on Image Processing*, 31:4719–4732, 2022. 2, 5, 6

[46] Jian Yao, Marko Boben, Sanja Fidler, and Raquel Urtasun. Real-time coarse-to-fine topologically preserving segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2015. 1, 2, 3, 5, 6, 7

[47] Donghun Yeo, Jeany Son, Bohyung Han, and Joon Hee Han. Superpixel-based tracking-by-segmentation using markov chains. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2017. 1

[48] Ye Yuan, Wei Zhang, Hai Yu, and Zhiliang Zhu. Superpixels with content-adaptive criteria. *IEEE Transactions on Image Processing*, 30:7702–7716, 2021. 5

[49] Yuhang Zhang, Richard Hartley, John Mashford, and Stewart Burn. Superpixels via pseudo-boolean optimization. In *IEEE International Conference on Computer Vision*, 2011. 1, 2, 6

[50] Meng Zhou, Zhe Xu, Kang Zhou, and Raymond Kai-yu Tong. Weakly supervised medical image segmentation via superpixel-guided scribble walking and class-wise contrastive regularization. In *Medical Image Computing and Computer Assisted Intervention*. 2023. 1, 2

[51] Pei Zhou, Xuejing Kang, and Anlong Ming. Vine spread for superpixel segmentation. *IEEE Transactions on Image Processing*, 32:878–891, 2023. 1, 2, 5, 6

[52] Alex Zihao Zhu, Jieru Mei, Siyuan Qiao, Hang Yan, Yukun Zhu, Liang-Chieh Chen, and Henrik Kretzschmar. Superpixel transformers for efficient semantic segmentation. In *IEEE International Conference on Intelligent Robots and Systems*, 2023. 1