Multiscale Superpixels and Supervoxels Based on Hierarchical Edge-Weighted Centroidal Voronoi Tessellation

Youjie Zhou[1], Lili Ju[2] and Song Wang[1]
[1]Department of Computer Science & Engineering, University of South Carolina, SC 29208
[2]Department of Mathematics, University of South Carolina, SC 29208
zhou42@email.sc.edu, ju@math.sc.edu, songwang@cec.sc.edu

Abstract

Superpixels and supervoxels play an important role in many computer vision applications, such as image segmentation, object recognition and video analysis. In this paper, we propose a hierarchical edge-weighted centroidal Voronoi tessellation (HEWCVT) method for generating superpixels/supervoxels in multiple scales. In this method we model the problem as a multilevel clustering process: superpixels/supervoxels in one level are clustered to obtain larger size superpixels/supervoxels in the next level. In the finest scale, the initial clustering is directly conducted on pixels/voxels. The clustering energy involves both color/feature similarities and the proposed boundary smoothness of superpixels/supervoxels. The resulting superpixels/supervoxels can be easily represented by a hierarchical tree which describes the nesting relation of superpixels/supervoxels across different scales. We evaluate and compare the proposed method with several state-of-the-art superpixel/supervoxel methods on standard image and video datasets. Both quantitative and qualitative results show that the proposed HEWCVT method achieves superior or comparable performances to other methods.

1. Introduction

By compactly representing 2D/3D images using a collection of perceptually meaningful atomic regions/volumes, superpixels/supervoxels have become a standard tool in many vision applications [8, 15, 13, 17]. Good superpixels/supervoxels usually have several desired properties [12, 23]: 1) all the pixels/voxels in a superpixel/supervoxel share similar features, such as color and/or texture; 2) all the generated superpixels/supervoxels are compact and uniformly distributed; 3) the superpixel/supervoxel boundaries well align with structural boundaries in the original image.

In addition, many vision applications require the use of multiscale superpixels/supervoxels with different coarse levels to better infer the high-level structural information [7, 11, 13, 17, 24, 22, 9]. Multiscale superpixels/supervoxels can usually be obtained by varying certain configurations, such as the number of or the average size of superpixels/supervoxels. However, simply varying these configurations may not generate multiscale superpixels/supervoxels with boundary consistency, i.e., the boundaries in a coarser level may not be drawn from the boundaries in a finer level. This way, the superpixels/supervoxels in different scales may not show a hierarchical nested relations, which is important for inferring high-level structural information [11, 5, 17, 7, 13].

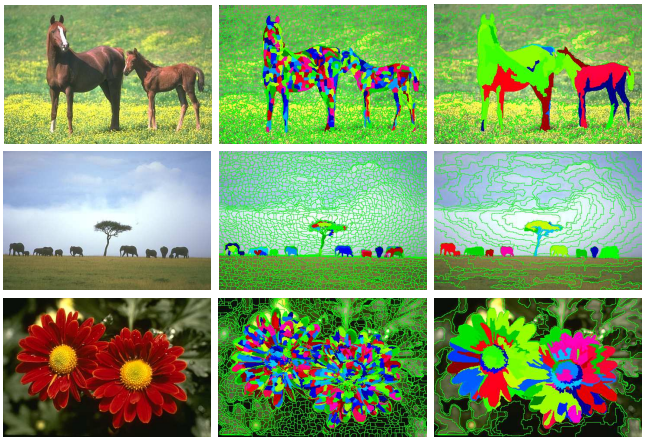


Figure 1: Sample results of the proposed HEWCVT method on 2D images: the original image (left column), superpixels constructed on the finest level of hierarchy (middle column) and superpixels constructed on the highest level of hierarchy (right column). Superpixels of objects are visualized by colorful patches. The hierarchical nested relations among superpixels can be easily verified.

In this paper, we develop a *hierarchical edge-weighted centroidal Voronoi tessellation* (HEWCVT) method for generating multiscale superpixels/supervoxels. In this method, superpixels/supervoxels in a finer level is clustered

978-1-4799-6683-7/15 $31.00 © 2015 IEEE
DOI 10.1109/WACV.2015.148

to achieve superpixels/supervoxels in a new coarser level. In the finest level, all the image pixels/voxels are taken as the entities for HEWCVT clustering. This iterative clustering process guarantees the hierarchical nested relations across different levels. In HEWCVT method, the clustering energy consists of not only a term that measures the color/feature between superpixels/supervoxels, but also a proposed edge term that measures the boundary smoothness of the obtained superpixels/supervoxels. With this edge term, the proposed HEWCVT method is able to produce superpixel/supervoxel boundaries better aligned with the underlying structural boundaries in each level. Examples of superpixels on 2D images are shown in Figure 1. In the experiments, we justify the proposed method by qualitatively and quantitatively comparing its performance with the performance of several other state-of-the-art superpixels/supervoxels methods on three standard image/video datasets.

1.1. Related work

In [12], the Turbopixel algorithm is developed for generating superpixels on 2D images by iteratively dilating seed locations using level-set based curve evolution. In each iteration it encourages regular space tessellation by using constraints based on local image gradients. However, it is nontrivial to extend Turbopixel to produce multiscale superpixels with hierarchical nested relations or generate 3D supervoxels from 3D images (e.g., videos). Furthermore, superpixels produced by Turbopixel are often scattered and noisy since it does not consider the boundary smoothness of superpixels. In [1], the *k*-means algorithm is extended to a SLIC algorithm for generating superpixels/supervoxels, by restricting the search space inside each superpixel/supervoxel and combining color and spatial proximity into a weighted distance function. SLIC is conducted only on pixels and may not guarantee the hierarchical nested relations across different scales. In [19], superpixels/supervoxels are generated by stitching overlapping patches such that each pixel belongs to only one of patches. This algorithm is built upon the GraphCut-based MRF energy optimization framework. However, our later experiments show that it is often difficult to catch structural boundaries when the number of superpixels is small.

More related to the proposed HEWCVT method is the VCells algorithm developed in [21]. VCells generates superpixels by using a modified edge-weighted centroidal Voronoi tessellation (EWCVT) model [20] on pixels. Starting from a spatially uniform tessellation of the image space, VCells iteratively moves the superpixel boundaries to better align with the structural boundaries. Except for minimizing an energy involving color/intensity similarity and boundary smoothness, VCells also enforces the connectivity of each superpixel in the optimization. However, as many other methods, VCells is only conducted on pixels and cannot produce multiscale superpixels with hierarchical nested relations. In the HEWCVT method, we propose a multiscale clustering algorithm, and a boundary smoothness measurement for superpixels/supervoxels.

Multiscale superpixels/supervoxels are usually obtained directly from multiscale 2D/3D image segmentation algorithms by varying certain configurations as discussed previously. Several multiscale image/video segmentation algorithms have been utilized in superpixel/supervoxel construction recently [16, 8, 23]. Both of them are based on graph aggregation using color/feature similarity and therefore the obtained superpixels/supervoxels are nested across different scale levels. However, without specific constraints on the superpixel/supervoxel connectivity and boundary smoothness, the resulting superpixels/supervoxels could be quite fragmented and scattered (Figure 3). We justify the performance of the proposed HEWCVT method by comparing with many of these state-of-the-art superpixels/supervoxels algorithms.

This paper is organized as follows. Section 2 reviews the EWCVT model and algorithms. We propose the HEWCVT model and develop algorithms for its construction in Section 3. Experiments and evaluations are presented in Section 4. Finally we give concluding remarks in Section 5.

2. Edge-Weighted Centroidal Voronoi Tessellation

Let $\mathbb{U} = \{\vec{u}(i,j) \mid (i,j) \in \mathbb{I}\}$ denote the set of color or feature vectors of a 2D image $\mathbb{I}$ (extension to 3D images will be discussed in Section 3.4), where $\vec{u}$ is the color/feature function associated with $\mathbb{I}$. In the experiments, we use the `Lab` color feature. For L arbitrary color vectors $\mathcal{W} = \{\vec{w}_l\}_{l=1}^{L}$ (called *generators*), the corresponding *Voronoi tessellation* of $\mathbb{U}$ is defined as $\mathcal{V} = \{V_l\}_{l=1}^{L}$ such that $V_l = \{\vec{u}(i,j) \in \mathbb{U} \mid \|\vec{u}(i,j) - \vec{w}_l\| < \|\vec{u}(i,j) - \vec{w}_m\|,\ m = 1,\ldots,L \text{ and } m \neq l\}$, where $\|\cdot\|$ is a distance function defined on $\mathbb{U}$. Given a weight or density function ρ defined on each pixel of $\mathbb{I}$, we can further define the centroid (i.e., the center of mass) of each Voronoi region V_l as $\vec{w}_l^*$ such that $\vec{w}_l^* = \min_{\vec{w} \in V_l} \sum_{\vec{u}(i,j) \in V_l} \rho(i,j)\|\vec{u}(i,j) - \vec{w}\|^2$.

If the generators $\{\vec{w}_l\}_{l=1}^{L}$ of the Voronoi regions $\{V_l\}_{l=1}^{L}$ of $\mathbb{U}$ are the same as their corresponding centroids, i.e.,

$$\vec{w}_l = \vec{w}_l^*,\ l = 1,\ldots L,$$

then we call the Voronoi tessellation $\{V_l\}_{l=1}^{L}$ a *centroidal Voronoi tessellation* (CVT) of $\mathbb{U}$. Since each Voronoi region V_l stands for a cluster in the color space we can easily construct a corresponding partition of the 2D image $\mathbb{I}$ using the correspondence between pixel indices and color vectors through $\vec{u}$. Let $\mathcal{C} = \{C_l\}_{l=1}^{L}$ denote a clustering of the physical space of the image $\mathbb{I}$, then the CVT clustering

energy can be defined as

$$E_{cvt}(\mathcal{C},\mathcal{W}) = \sum_{l=1}^{L} \sum_{(i,j)\in C_l} \rho(i,j)\|\vec{u}(i,j) - \vec{w}_l\|^2. \quad (1)$$

The construction of CVTs often can be viewed as a clustering energy minimization problem, i.e., solving $\min_{(\mathcal{C},\mathcal{W})} E_{cvt}(\mathcal{C},\mathcal{W})$. The Lloyd method [6] (equivalent to the weighted k-means) has been widely used to compute CVTs, which is basically iterations between constructing Voronoi regions and centroids. Assume that the Euclidean distance is used for the color space, then we simply have the centroid of the cluster C_l as $\vec{w}_l^* = \sum_{(i,j)\in C_l} \rho(i,j)\vec{u}(i,j) / \sum_{(i,j)\in C_l} \rho(i,j)$.

In order to enforce the smoothness of segment boundaries, a special edge energy was proposed and added into the clustering energy [20, 21]. Specifically, let us define an indicator function $\chi(i,j) : \mathbb{N}_\omega(i,j) \to \{0,1\}$ on the neighborhood of pixel (i,j) with radius ω as

$$\chi_{(i,j)}(i',j') = \begin{cases} 1 & if\ \pi(i',j') \neq \pi(i,j) \\ 0 & otherwise \end{cases}$$

where $\pi(i,j)$ tells the cluster index that (i,j) belongs to. Then the edge energy is defined as

$$E_{edge}(\mathcal{C}) = \sum_{(i,j)\in\mathbb{I}} \sum_{(i',j')\in\mathbb{N}_\omega(i,j)} \chi_{(i,j)}(i',j'). \quad (2)$$

Figure 2a illustrates the boundary smoothness measurement on a single pixel. It has been shown in [20] that $E_{edge}(\mathcal{C})$ is proportional to the total length of boundaries in $\mathcal{C}$ in the limit. Finally, the edge-weighted CVT clustering energy can be defined as

$$E_{ewcvt}(\mathcal{C},\mathcal{W}) = E_{cvt}(\mathcal{C},\mathcal{W}) + \lambda E_{edge}(\mathcal{W}) \quad (3)$$

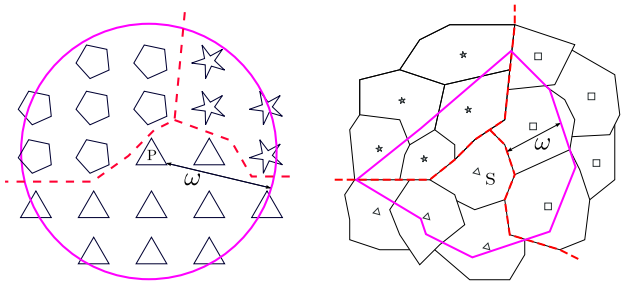
where λ is a weight parameter balancing the clustering energy and the edge energy. Construction of EWCVTs is equivalent to solving the minimization problem $\min_{(\mathcal{C},\mathcal{W})} E_{ewcvt}(\mathcal{C},\mathcal{W})$. An edge-weighted distance function from a pixel (i,j) to a cluster center (generator) $\vec{w}_k$ was derived for the energy E_{ewcvt} as

$$dist((i,j),\vec{w}_k) = \sqrt{\rho(i,j)\|\vec{u}(i,j) - \vec{w}_k\|^2 + 2\lambda\tilde{n}_k(i,j)} \quad (4)$$

where $\tilde{n}_k(i,j) = |\mathbb{N}_\omega(i,j)| - n_k(i,j) - 1$ with $n_k(i,j) = \sum_{(i',j')\in\mathbb{N}_\omega(i,j)} \pi(i',j') \neq k$. Based on the above distance function, some efficient algorithms for constructing EWCVTs are suggested in [20] based on the k-means type techniques. We specially note that a cluster C_l produced by the EWCVT method may consist of physically disconnected regions in the image $\mathbb{I}$, i.e., multiple segments

3. Hierarchical Edge-Weighted Centroidal Voronoi Tessellation

The proposed hierarchical method begins with an oversegmentation on pixels using the a modified EWCVT algorithm with strictly enforcing simple-connectivity of superpixels [21], which is taken as the finest level of superpixels in the hierarchy. For the higher levels, we merge finer level superpixels having similar color features, meanwhile preserve superpixel connectivity and enforce the smoothness of superpixels.



(a) Boundary smoothness measurement for a pixel P. Each pixel is visualized as a polygon and its shape stands for the pixel's current cluster assignment.

(b) Boundary smoothness measurement for a superpixel S. Each polygon represents a superpixel and the shape of its center marker stands for the superpixel's current cluster assignment.

Figure 2: Boundary smoothness measurement illustrations. Dash lines are cluster boundaries. Pink curve indicates the local neighborhood area for smoothness measurement.

3.1. Finest level

At the finest level we deal with generation of superpixels directly from the image pixels. Let M_1 be the desired number of superpixels. Similar to the VCells algorithm proposed in [21], we first use the classic k-means with the Euclidean norm on pixel coordinates $\mathbb{I}$, to generate M_1 simply-connected and quasi-uniformly distributed superpixels for the input image. We also set $\rho \equiv 1$ here. Next we apply the VCells algorithm to the initial superpixel configuration where we only allow transferring of boundary pixels between neighbor clusters at each iteration. The whole algorithm is described in Algorithm 1. If $\pi(i,j)$ is different from its 4 neighbors, i.e., $(i \pm 1, j)$ or $(i, j \pm 1)$, we say (i,j) is a boundary pixel, and denote $\mathcal{B}$ as the set of all boundary pixels. We remark that each pixel moving between neighbor clusters in Algorithm 1 will decrease the energy E_{ewcvt}, thus Algorithm 1 guarantees monotonic decreasing of E_{ewcvt} along the iterations till it terminates.

We note that there is still no guarantee for exact preservation of the simple-connectivity property in the algorithm above. Thus in the end we perform a filtering step to further enforce the simple-connectivity of superpixels, which is widely used in other superpixel algorithms [1, 12, 19, 21] and will be described in Section 3.3.

3.2. Higher levels

At a higher level q ($q > 1$), assume that we have a superpixel from the previous level $q-1$, $\mathbb{S} = \{S_m\}_{m=1}^{M_{q-1}}$. Given

Algorithm 1 (Finest-Level Superpixel Algorithm)

Input: The target 2D image $\mathbb{I}$ and the color function $\vec{u}$
M_1: Number of desired superpixels
$niter$: Maximum number of iterations

0 **Initialization**: Construct the initial superpixels of $\mathbb{I}$, $\{C_l\}_{l=1}^{M_1}$ using the *k*-means with the Euclidean distance and the feature function $\vec{u}^+$.
1 **FOR** each $C_k \in \{C_l\}_{l=1}^{M_1}$
2 Compute centroid $\vec{w_k} = \frac{1}{|C_k|} \sum_{(i,j)\in C_k} \vec{u}(i,j)$
3 **FOR** $iter = 1$ to $niter$
4 Create the set of boundary pixels $\mathcal{B}$
5 **FOR** each $(i,j) \in \mathcal{B}$
6 Find the closest centroid to the pixel (i,j)
$\vec{w}_k \in \{\vec{w}_l \mid l \in \pi(\mathcal{N}_4(i,j))\}$
w.r.t. the edge-weighted distance (Eq. (4))
7 **IF** $\pi(i,j) \neq k$
8 Set $\tilde{k} = \pi(i,j)$ and $\pi(i,j) = k$
9 Update $\vec{w}_k$, $\vec{w}_{\tilde{k}}$
10 **IF** there is no cluster index change
11 Break
12 Perform the simple-connectivity filtering

Output: The cluster/superpixel index function π

the desired number of superpixels M_q ($M_q < M_{q-1}$) in Level q, we will merge adjacent superpixels to reach that goal according to minimization of certain energy function. Each superpixel is treated as a point and we will cluster them into M_q simply-connected parts. Thus we can easily build a tree structure for superpixels for these two levels.

The initialization step is different from that in the finest level. The most intuitive idea is to apply the *k*-means clustering on the set of average coordinates of all superpixels constructed in the previous level. However, the merged superpixels may not be simply-connected. Instead, we first build a superpixel graph $G = (V, E, \mathcal{E})$, where V consists of all the previous level's superpixels $\{S_m\}_{m=1}^{M_{q-1}}$ and E is the set of all pairs of neighbor superpixels. The edge weight for $(S_a, S_b) \in E$ is defined as

$$\mathcal{E}(S_a, S_b) = \frac{\|\vec{u}(S_a) - \vec{u}(S_b)\|}{\max_{(S_a,S_b)\in E} \|\vec{u}(S_a) - \vec{u}(S_b)\|} \tag{5}$$

where $\vec{u}(S) = \frac{1}{|S|}\sum_{(i,j)\in S} \vec{u}(i,j)$ denotes the average color vector of all the pixels belonging to the superpixel S. Then the superpixel graph G will be partitioned into M_q subgraphs which are considered as initialized superpixels at level q. The proposed HEWCVT method will refine initialized superpixels later. Therefore, any graph partition algorithm can be used for this initialization. Based on algorithm efficiency and code availability, we choose the METIS algorithm [10] here.

We define the density function ρ on $\mathbb{S}$ as $\rho(S_m) = |S_m|$, i.e., the number of pixels contained in the superpixel $S_m \in \mathbb{S}$. Let $\mathcal{C}^{sp} = \{C_l^{sp}\}_{l=1}^{M_q}$ be a clustering of $\mathbb{S}$ and $\mathcal{W} = \{\vec{w}_l\}_{l=1}^{M_q}$ be an arbitrary set of color vectors. Then we define the new CVT clustering energy as

$$E_{cvt-sp}(\mathcal{C}^{sp}, \mathcal{W}) = \sum_{l=1}^{M_q} \sum_{S\in C_l^{sp}} \rho(S)\|\vec{u}(S) - \vec{w}_l\|^2. \tag{6}$$

In order to measure the boundary length (or the smoothness) of superpixels, we propose an edge energy for the superpixel image. As illustrated in Figure 2b, we define the local neighborhood $\mathbb{N}_\omega(S)$ for a superpixel $S \in \mathbb{S}$ as

$$\mathbb{N}_\omega(S) = \bigcup_{(i,j)\in\mathcal{B}(S)} \mathbb{N}_\omega(i,j) - S$$

where $\mathcal{B}(S)$ denotes the set of all boundary pixels of the superpixel S. Then we define the edge energy as

$$E_{edge-sp}(\mathcal{C}^{sp}) = \sum_{S\in\mathbb{S}} \sum_{(i,j)\in\mathbb{N}_\omega(S)} \Gamma_S(i,j) \tag{7}$$

where $\Gamma_S(i,j) : \mathbb{N}_\omega(S) \to \{0,1\}$ is an indicator function, similar as $\chi(i,j)$ in Eq. 2, and is defined by

$$\Gamma_S(i,j) = \begin{cases} 1 & if\ \pi(i,j) \neq \pi(S) \\ 0 & otherwise \end{cases}$$

where $\pi(S)$ returns the cluster index of the superpixel S in $\mathcal{C}^{sp}$.

Finally, the edge-weighted CVT clustering energy for superpixels can be defined as

$$E_{ewcvt-sp}(\mathcal{C}^{sp}, \mathcal{W}) = E_{cvt-sp}(\mathcal{C}^{sp}, \mathcal{W}) + \lambda E_{edge-sp}(\mathcal{C}^{sp}). \tag{8}$$

We can derive the distance from a superpixel S to a cluster center $\vec{w_k}$ corresponding to the above energy as

$$dist(S, \vec{w}_k) = \sqrt{\rho(S) \parallel \vec{u}(S) - \vec{w}_k \parallel^2 + 2\lambda\tilde{n}_k(S)} \tag{9}$$

where $\tilde{n}_k(S)$ measures the number of inconsistent pixels in the neighborhood of the superpixel S: $\tilde{n}_k(S) = |\mathbb{N}_\omega(S)| - n_k(S)$ with $n_k(S) = \sum_{(i,j)\in\mathbb{N}_\omega(S)} \pi(i,j) \neq k$.

Furthermore, in order to keep superpixels simply connected, we follow the idea in the finest level (Section 3.1), i.e., only superpixels located at cluster boundaries will be considered during the clustering, and we only allow cluster index change among adjacent clusters. The whole algorithm is described in Algorithm 2. We again remark that similar to Algorithm 1, Algorithm 2 guarantees monotonic decreasing of $E_{ewcvt-sp}$ along the iterations till it terminates.

3.3. Simple-connectivity filtering

Although we have enforced that the pixel/superpixel transferring can only occur among adjacent clusters, due to the image noises, few superpixels may still break into several disconnected parts and/or contain holes (especially in 3D cases). Thus after the HEWCVT clustering process, we merge small ($|S| \leq \varepsilon$) and isolated superpixels into their surroundings. Similar post-step has been applied in several state-of-the-art superpixel/supervoxel methods [1, 12, 19, 21].

Algorithm 2 (Higher Level Superpixel Merging Algorithm)

Input: The superpixel image $\mathbb{S}$ and the color function $\vec{u}$
M_q: Number of desired superpixels
$niter$: Maximum number of iterations

Initialization: Construct the superpixel graph G and partition $\mathbb{S}$ into M_q simply-connected regions $\{C_l^{sp}\}_{l=1}^{M_q}$ using METIS [10]

FOR each $C_k^{sp} \in \{C_l^{sp}\}_{l=1}^{M_q}$

  Compute centroid $\vec{w}_k = \frac{\sum_{S \in C_k^{sp}} \rho(S)\vec{u}(S)}{\sum_{S \in C_k^{sp}} \rho(S)}$

FOR $iter = 1$ to $niter$

  Create the set of boundary superpixels $\mathcal{B}(\mathbb{S})$

  FOR each $S \in \mathcal{B}(\mathbb{S})$

    Find the closest centroid to S
    $\vec{w}_k \in \{\vec{w}_l \mid l \in \pi(\mathcal{N}(S)\}$
    w.r.t. the edge-weighted distance (Eq. (9))

    IF $\pi(S) \neq k$

      Set $\tilde{k} = \pi(S)$ and $\pi(S) = k$

      Update $\vec{w}_k$, $\vec{w}_{\tilde{k}}$

  IF there is no cluster index change

    Break

Perform the simple-connectivity filtering

Output: The cluster/superpixel index function π

Specifically, there are two cases: 1) in the finest level, for each pixel p in a small or isolated superpixel S, we first locate its nearest neighbor pixel p' in surrounding superpixels S' and then merge p into S'; 2) in the higher levels, we can associate each pixel p in S with a surrounding superpixels as in the finest level, and this association can be viewed as a vote from p to a surrounding superpixel. We merge S into S' that has the majority vote.

3.4. Extension to supervoxels

We can easily extend the proposed hierarchical method into 3D case. The major difference is the neighbor system among voxels and supervoxels. Instead of 4-neighbor system in 2D case, we use 6-neighborhood for the voxel level oversegmentation.

Another issue is that in the 2D case we assume the units of all coordinate directions are the same. For 3D images this assumption is still valid in most situations. However, for video data, the unit of the temporal direction could be different from those of spatial axises. Therefore for video data, one could use $\mathbb{I}_{3D} = H * (i, j, k)^T$ where $H = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & \gamma_k \end{bmatrix}$ is a scaling matrix and γ_k is data dependent. In the video experiments, we just simply used $H = I_{3\times 3}$ and it worked fine for the test video data.

3.5. Adaptive determination of λ

The energy weight parameter λ defined in Eqs. (3) and (8) balances the ratio between the CVT clustering energy E_{cvt} (or E_{cvt-sp}) and the edge energy E_{edge} (or $E_{edge-sp}$). However, these energies are varying from different images/videos and/or change along different scale levels. Especially in video segmentation, different videos also have variant number of frames and frame rates. Thus a fixed λ is obviously inappropriate. Instead we aim at controlling the ratio between E_{cvt} and λE_{edge}. Therefore, given an predetermined energy ratio θ that $\frac{E_{cvt}}{\lambda E_{edge}} = \frac{E_{cvt-sp}}{\lambda E_{edge-sp}} = \theta$, we can adjust λ adaptively by setting $\lambda^{(iter)} = \frac{E_{cvt}^{(iter-1)}}{\theta E_{edge}^{(iter-1)}}$ at each iteration in Algorithm 1 and $\lambda^{(iter)} = \frac{E_{cvt-sp}^{(iter-1)}}{\theta E_{edge-sp}^{(iter-1)}}$ in Algorithm 2.

4. Experiments

We tested the proposed HEWCVT method on three standard image/video segmentation benchmarks which have been widely used for evaluating the performance of superpixels/supervoxels: 1) the Berkeley dataset [14], which consists of 300 color images (481×321 or 321×481); 2) the Weizmann dataset [2], which consists of 200 color images of size approximately 300×225; 3) the Chen video dataset [3], which consists of 8 color videos of approximately 85 frames (240×160) each. For the evaluation metrics we take the three widely used standard superpixel/supervoxel measurements, including boundary recall, under-segmentation error and segmentation accuracy [1, 19, 23]. For each of the three metrics we report the average values on each dataset.

On the first two datasets we compared the performance of the proposed superpixel method with the algorithm (GraphCut) of [19], the SLIC algorithm [1], and the VCells algorithm [21]. We do not include other superpixel algorithms such as the Turbopixel algorithm [12], and other segmentation based methods such as the NormalizedCut algorithm [15], Meanshift [4] and Quickshift [18] algorithms, into our comparison because: according to [1] SLIC outperforms many state-of-the-art superpixel and segmentation algorithms on the Berkeley dataset and we have included SLIC into our comparison. On the third dataset we compared the proposed supervoxel method with the GBH algorithm [8] and the SWA algorithm [16].

4.1. Implementation details

We implemented the proposed method and the evaluation algorithm in C/C++. For the comparison algorithms, we used implementations published by their authors. The video benchmark evaluation code is based on [23]. All experiments were conducted on an Intel Xeon 2.4GHz processor.

	# of superpixels in each level				
	2048	1024	512	256	128
BSDS	1.5	500	1000	1000	1000
W1	1.65	210	510	210	1000
W2	4	100	100	100	100

Table 1: Grid search results on θ for three image data sets: the Berkeley dataset (BSDS), the Weizmann dataset with 1 object (W1) and the Weizmann dataset with 2 objects (W2).

There are two parameters in the proposed method: the neighborhood radius ω and the energy ratio θ. We set $\omega = 2$ for all the levels in the hierarchy for both image and video processing. For θ, we did grid search on a validation set (30% of the image data), at all levels, and picked a θ value at each level that gave the best quantitative performance. Then we applied these values of θ to all the remaining images in the datasets. We did the same grid search for VCells. For each image data set, we list θs at each level in Table 1. We didn't perform grid search in the video experiments and θ was set to be 1 for all videos on all levels.

We set $\epsilon = 5$ and $\epsilon = 15$ to filter out noisy superpixels and supervoxels respectively. In our experiments, we found HEWCVT actually produced very few noisy superpixels/supervoxels (less than 1% of superpixels/supervoxels are filtered as noises).

Similar to the supervoxel benchmark in [23], we report the average performance in terms of the number of the constructed supervoxels, range from 100 to 3000 with a step of 100. In the proposed method, different number of supervoxels are obtained in different levels of hierarchy. However, two comparison methods, GBH and SWA, do not enforce connectivity of each supervoxel in its result. An example is shown in Figure 3 – GBH generates 35 supervoxels on a video and these 35 supervoxels actually consist of 15226 connected components. *In [23], performance of such a result is considered to be for* 35 *supervoxels and this is clearly unfair and inappropriately favors GBH and SWA in the performance evaluation.* For a fairer comparison, we applied the same filter $\epsilon = 15$ to remove such fragments and then count each connected component as a separate supervoxel in evaluating GBH and SWA in this paper. *Therefore, the performance curves of GBH and SWA reported below are different from those reported in [23].* We include the original performance curves in the supplemental material.

4.2. Evaluation results

In order to quantitatively evaluate the performance of superpixels/supervoxels, we used human-labeled segmentation as the ground truth because the superpixel/supervoxel boundaries should well align with structural boundaries. We applied three standard superpixel/supervoxel measurements: boundary recall, under-segmentation error and segmentation accuracy [1, 19, 23].

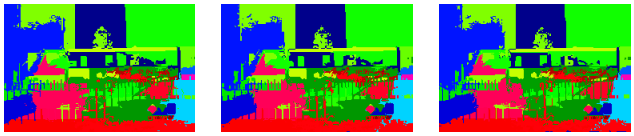


Figure 3: An illustration of the disconnectivity issue in GBH. Each supervoxel (with a specific color) from GBH actually contains many disjoint fragments.

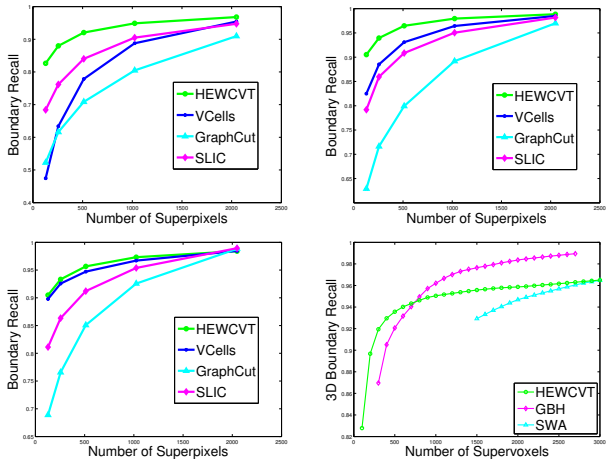


Figure 4: Boundary recall on the Berkeley dataset (top-left), the Weizmann dataset with 1 object (top-right), the Weizmann dataset with 2 objects (bottom-left) and the Chen video dataset (bottom-right). Higher is better.

Boundary Recall – Given a structural boundary, we search for a boundary in the constructed superpixels/supervoxels within a t pixels/voxels distance. In experiments, we set $t = 2$ for both images and videos. In general the larger the number of superpixels/supervoxels, the more boundaries, and the better the boundary recall. The boundary recall for different datasets are shown in Figure 4. We can see that, in constructing superpixels from images, HEWCVT clearly achieves better boundary recall than other state-of-the-art methods, and in constructing supervoxels from videos, HEWCVT achieves comparable performance to GBH and better performance than SWA. When the number of supervoxels is very large, supervoxels generated by GBH become highly scattered with a large number of disconnected fragments. Therefore GBH achieves better boundary recall. However, highly scattered supervoxels lead to lower accuracy in catching structural boundaries, which is measured by other two metrics below.

Undersegment Error – This metric measures the fraction of superpixels/supervoxels that is leaked across the boundary of the ground-truth segments (detailed formula

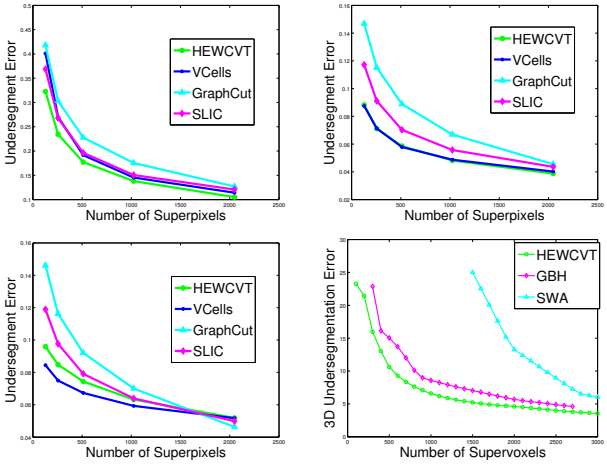


Figure 5: Undersegment error on the Berkeley dataset (top-left), the Weizmann dataset with 1 object (top-right), the Weizmann dataset with 2 objects (bottom-left) and the Chen video dataset (bottom-right). Lower is better.

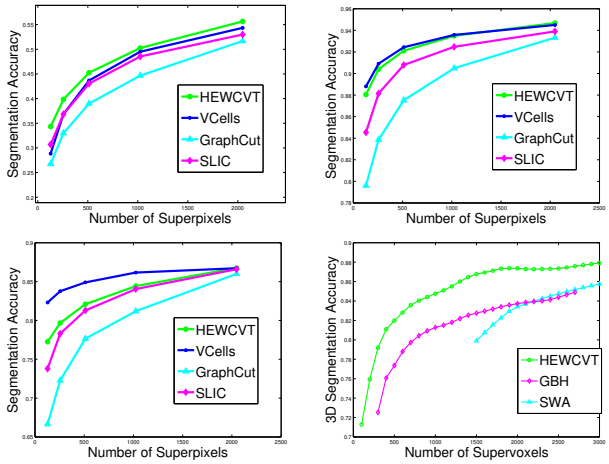


Figure 6: Segmentation accuracy on the Berkeley dataset (top-left), the Weizmann dataset with 1 object (top-right), the Weizmann dataset with 2 objects (bottom-left) and the Chen video dataset (bottom-right). Higher is better.

and setting follow [1]). Figure 5 shows the results of undersegment error on different datasets. HEWCVT performs better than all the comparison methods on the Berkeley dataset and achieves a comparable performance with VCells on the Weizmann datasets, and on the video dataset, HEWCVT outperforms the two comparison methods.

Segmentation Accuracy – This metric is contrary to the undersegment error. It measures the fraction of a ground-truth segment that is correctly recovered by the constructed superpixels/supervoxels (detailed formula and setting follow [23]). Figure 6 reports results of the segmentation accuracy on different datasets. HEWCVT outperforms all the comparison methods on the Berkeley dataset and achieves a comparable performance with VCells on the Weizmann datasets, and on the video dataset, HEWCVT performs better than GBH and SWA.

Superpixels/supervoxels constructed from sample images and videos are shown in Figures 7 and 8 respectively. In Figure 7, odd columns show the superpixels in the finest scale while even columns show the superpixels in the coarsest scale. We can see that, compared with the three comparison methods, HEWCVT produces more uniform superpixels in the finest scale while catches structural boundaries more accurately in the coarsest scale. From Figure 8, we can see that, with a similar number of supervoxels, HEWCVT can produce more uniform supervoxels to catch the structural boundaries, but without generating many small fragments, when compared with GBH and SWA.

5. Conclusions

We have proposed a hierarchical edge-weighted centroidal Voronoi tessellation method for generating multi-scale superpixels/supervoxels. The structural boundaries are consistent among superpixels/supervoxels in different levels. The method is easier to implement and the running times on images/videos are fast. Quantitative and qualitative results from various experiments show that the HEWCVT method can achieve superior or comparable performances over several current state-of-the-art algorithms. For supervoxel generation in videos, we will further investigate utilization of motion-involved features in the energy functions for supervoxel merging and also extend the proposed algorithm to handle streaming videos.

Acknowledgment This work was supported by AFOSR FA9550-11-1-0327, NSF IIS-1017199 and was partially supported by the Open Project Program of the State Key Lab of CAD & CG (Grant No. A1420) in Zhejiang University, China.

References

[1] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Süsstrunk. SLIC superpixels compared to state-of-the-art superpixel methods. *TPAMI*, 34:2274–2282, 2012.

[2] S. Alpert, M. Galun, R. Basri, and A. Brandt. Image segmentation by probabilistic bottom-up aggregation and cue integration. In *CVPR*, 2007.

[3] A. Chen and J. Corso. Propagating multi-class pixel labels throughout video frames. In *Procs of Western New York Image Processing Workshop*, 2010.

[4] D. Comaniciu and P. Meer. Mean shift: A robust approach toward feature space analysis. *TPAMI*, 24:603–619, 2002.

[5] S. Dickinson, A. Levinshtein, and C. Sminchisescu. Perceptual grouping using superpixels. *Pattern Recognition*, 7329:13–22, 2012.

[6] Q. Du, M. Gunzburger, and L. Ju. Advances in studies and applications of centroidal voronoi tessellations. *Numer. Math. Theo. Meth. Appl.*, 3:1–24, 2010.

[7] S. Gould, J. Rodgers, D. Cohen, G. Elidan, and D. Koller. Multi-class segmentation with relative location prior. *IJCV*, 80:300–316, 2008.

[8] M. Grundmann, V. Kwatra, M. Han, and I. Essa. Efficient hierarchical graph-based video segmentation. In *CVPR*, 2010.

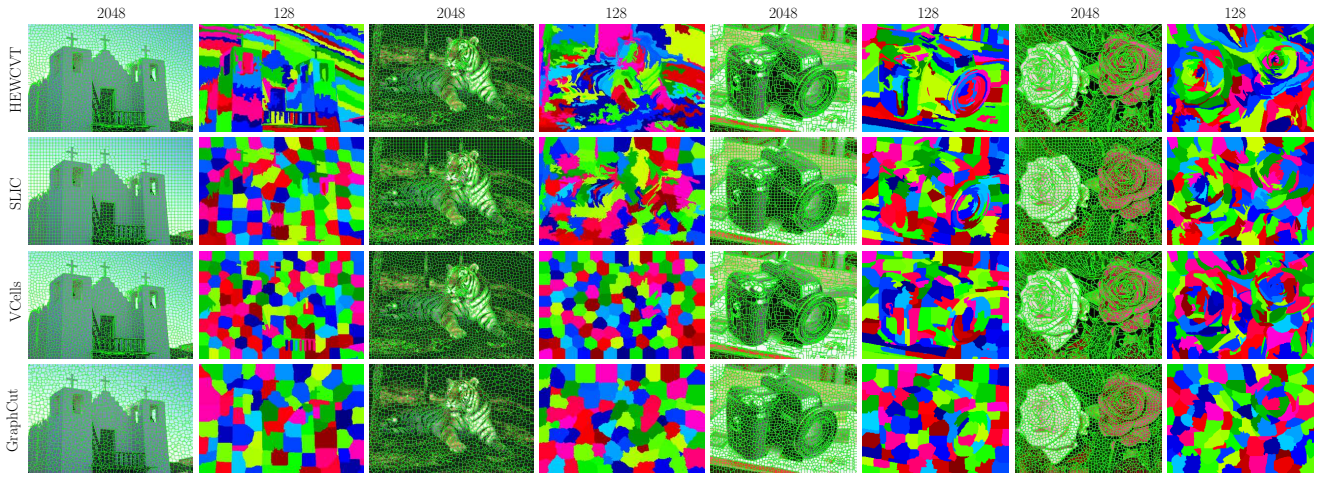


Figure 7: Qualitative comparisons of the four superpixel methods (HEWCVT, SLIC, VCells, GraphCut) on four images. The numbers at the top indicate the desired number of superpixels. Neighboring superpixels are shown in different color.

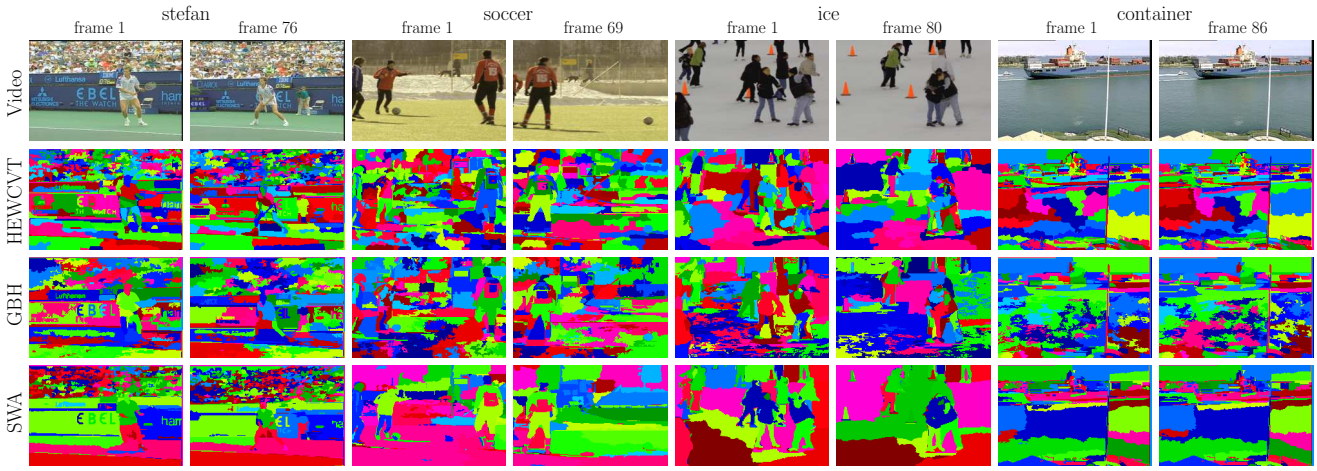


Figure 8: Qualitative comparisons of the three supervoxel methods (HEWCVT, GBH, SWA) on four videos. On each video, the number of supervoxels generated by these three methods are similar for fairer comparison. Neighboring supervoxels are shown in different color.

[9] X. He, R. S. Zemel, and D. Ray. Learning and incorporating top-down cues in image segmentation. In *ECCV*, 2006.

[10] G. Karypis and V. Kumar. A fast and high quality multilevel scheme for partitioning irregular graphs. *SIAM J. Sci. Comput.*, 20:359–392, 1998.

[11] P. Kohli, L. Ladický, and P. H. Torr. Robust higher order potentials for enforcing label consistency. *IJCV*, 82:302–324, 2009.

[12] A. Levinshtein, A. Stere, K. Kutulakos, D. Fleet, S. Dickinson, and K. Siddiqi. Turbopixels: Fast superpixels using geometric flows. *TPAMI*, 31:2290–2297, 2009.

[13] Z. Li, X.-M. Wu, and S.-F. Chang. Segmentation using superpixels: A bipartite graph partitioning approach. In *CVPR*, 2012.

[14] D. Martin, C. Fowlkes, D. Tal, and J. Malik. A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In *ICCV*, 2001.

[15] X. Ren and J. Malik. Learning a classification model for segmentation. In *ICCV*, 2003.

[16] E. Sharon, A. Brandt, and R. Basri. Fast multiscale image segmentation. In *CVPR*, 2000.

[17] A. Vazquez-Reina, S. Avidan, H. Pfister, and E. Miller. Multiple hypothesis video segmentation from superpixel flows. In *ECCV*, 2010.

[18] A. Vedaldi and S. Soatto. Quick shift and kernel methods for mode seeking. In *ECCV*, 2008.

[19] O. Veksler, Y. Boykov, and P. Mehrani. Superpixels and supervoxels in an energy optimization framework. In *ECCV*, 2010.

[20] J. Wang, L. Ju, and X. Wang. An edge-weighted centroidal voronoi tessellation model for image segmentation. *TIP*, 18:1844–1858, 2009.

[21] J. Wang and X. Wang. Vcells: Simple and efficient superpixels using edge-weighted centroidal voronoi tessellations. *TPAMI*, 34:1241–1247, 2012.

[22] S. Wang, H. Lu, F. Yang, and M.-H. Yang. Superpixel tracking. In *ICCV*, 2011.

[23] C. Xu and J. J. Corso. Evaluation of super-voxel methods for early video processing. In *CVPR*, 2012.

[24] Y. Yang, S. Hallman, D. Ramanan, and C. Fowlkes. Layered object detection for multi-class segmentation. In *CVPR*, 2010.