Unsupervised Scene Segmentation Using Sparse Coding Context

Yen-Cheng Liu · Hwann-Tzong Chen

Finished: 6 September 2011

Abstract This paper presents an approach to image understanding on the aspect of unsupervised scene segmentation. With the goal of image understanding in mind, we consider 'unsupervised scene segmentation' a task of dividing a given image into semantically meaningful regions without using annotation or other human-labeled information. We seek to investigate how well an algorithm can achieve at partitioning an image with limited human-involved learning procedures. Specifically, we are interested in developing an unsupervised segmentation algorithm that only relies on the contextual prior learned from a set of images. Our algorithm incorporates a small set of images that are similar to the input image in their scene structures. We use the sparse coding technique to analyze the appearance of this set of images; the effectiveness of sparse coding allows us to derive *a priori* the context of the scene from the set of images. Gaussian mixture models can then be constructed for different parts of the input image based on the sparse-coding contextual prior, and can be combined into an MRF-based segmentation process. The experimental results show that our unsupervised segmentation algorithm is able to partition an image into semantic regions, such as buildings, roads, trees, and skies, without using human-annotated information. The semantic regions generated by our algorithm can be useful, as pre-processed inputs for subsequent classification-based labeling algorithms, in achieving automatic scene annotation and scene parsing.

Keywords Unsupervised Image Segmentation · Semantic Scene Analysis · Sparse Coding · Markov Random Fields

Yen-Cheng Liu
Department of Computer Science
National Tsing Hua University, Taiwan

Hwann-Tzong Chen
Department of Computer Science
National Tsing Hua University, Taiwan
Tel.: +886-3-5731309
E-mail: htchen@cs.nthu.edu.tw

1 Introduction

One of the goals of computer vision is to develop algorithms and systems that understand the contents of images. To achieve this goal various attempts have been made through solving the related problems at different levels. For a general-purpose computer-vision system that can perform as well as or even better than humans, it might need to have the capabilities to detect and group low-level features like color, texture, or edge, to recognize various categories of objects, and to infer the relation and context of the objects, *etc.* Not only is solving the specific vision problem at each level difficult, but the dependencies on good solutions to different problems and the requirements for solving them simultaneously make developing a general computer-vision system still a challenging goal to pursue [6], [26].

Rather than searching for a general solution, this paper presents an approach to image understanding on the aspect of unsupervised scene segmentation. We seek to investigate how well an algorithm can achieve at partitioning an image with limited human-involved learning procedures. More specifically, the task of unsupervised scene segmentation is to divide a given image into semantically meaningful regions without using annotation or other human-labeled information. As far as the unsupervised setting is concerned, our work is similar to Russell *et al.* [19], where they use a large image database to extract information about regions from images of similar scenes and no further ground-truth labeling is required. Large image databases like the LabelMe database [22] or community photo collections have been shown to be useful for many vision applications, *e.g.*, image completion [4], scene parsing, object discovering and annotation [10], [15], [20], scene segmentation [19], [24], and building 3D scenes [21], [25]. Our work follows this line of thought, and is aimed to design a more efficient algorithm that solves image segmentation using photo collections.

Unsupervised image segmentation algorithms such as Normalized Cuts [23], Mean Shift [2], or Efficient Graph-Based Image Segmentation [3] have been widely used in computer vision. Nevertheless, they are more suitable for providing over-segmented regions or superpixels [5], [13]. To generate object-level regions would require further processing. For example, GrabCut [17] incorporates the hints provided by user into graph-cut optimization [1] to produce high-quality figure/ground segmentation. When the ground-truth user annotation of a large image dataset is available, the nonparametric scene parsing technique proposed by Liu *et al.* [10] can be used to find dense scene correspondences, and then transfer the labels from annotated images to a new input image.

Even without user annotations, collections of images may still be effective enough to provide useful information about segmentation if different images in a collection exhibit certain level of consistency or similarity. Simon and Seitz [24] present a segmentation method that can identify and segment interesting objects in a scene. Their approach applies a field-of-view constraint among images in the photo collections, and uses SIFT-based feature matching [11] to get the cue of feature co-occurrence for region labeling. The effectiveness of the method of [24] is owing to the abundance of co-occurred image features in different images. However, for images of regular scenes or non-tourist sites, finding corresponding images containing same buildings or similar objects would be improbable. Likewise, the setting of co-segmentation [9], [18] also makes use of the similarity in appearance among multiple images that include the same objects.

In addition to exploiting the similarity in objects' appearance, the similarity in scene structure and geometry can also be used to infer region labels. Russell *et al.* [19] present an unsupervised scene segmentation algorithm that extracts the boundary and region information using an image stack associated with an input image. Given an input image, their algorithm first searches from a very large dataset to collect several thousands of images that present a similar scene structure as the input image. This step is done by using the *gist* descriptor [14] to match two images according to their scene structures. The image stack entails data-driven boundary detection and region grouping, and the resulting boundary and region cues are combined in a Markov-random-field (MRF) optimization scheme to produce scene segmentation.

Although images with similar gist descriptors might also have similar scene structures, the appearance of their contents could widely differ from one another. We propose to use the sparse coding technique presented in [12] to learn a sparse dictionary of Gabor filters. The effectiveness of sparse coding allows us to employ only a small number of images for deriving *a priori* the context of the scene. Therefore, unlike the method of [19] which requires thousands of images and high computational cost to do boundary detection and region grouping, our algorithm can build the contextual prior of the scene from no more than a hundred images. Based on the contextual prior of the scene, we then construct Gaussian mixture models for different parts of the image, and combine the prior into MRF-based segmentation.

2 Overview of the Proposed Approach

Given an image of a scene, the human visual system can easily partition the image into semantically meaningful regions. This ability is partly due to the prior knowledge about objects in the scene, such as buildings, skies, roads, cars, and trees. To build a computer vision system that can recognize multiple categories of objects often requires a large number of human-annotated object images for training. On the other hand, the context information of a scene is also helpful in partitioning an image. For example, in an image of a normal scene, the ground and roads are usually at the bottom of the image, buildings are on the ground, cars may be adjacent to roads or buildings, and the sky is probably at the top of the image, etc. In this work we try to address the scene segmentation problem without using human-annotated information. Our approach is to exploit the context information. We develop an unsupervised segmentation algorithm that only relies on the contextual prior learned from a small set of images.

The nonparametric approach proposed by Russell *et al.* [19] is able to achieve good segmentation results via analyzing a stack of reference images that have similar scene structures. Their algorithm uses a large number of reference images ($5,000$ for each query in their cases), and the computation of extracting the region and boundary information from the image stack is very time-consuming, which may take more than an hour on a normal PC. We attempt to build a smaller-sized image stack, say a hundred images, for extracting useful context information. Furthermore, we seek to develop a more efficient way to obtain a contextual prior based on the sparse coding techniques.

In the next two sections, we first describe how to estimate the input image's contextual prior from images of similar scene structures. After estimating the contextual prior, we build Gaussian mixture models (GMMs) based on the contextual prior to model the color of the input image, as explained in section 5. A Markov-random-field (MRF) based energy-minimization problem is then constructed according to the GMMs, and is solved by graph-cuts. In section 6, an additional improvement stage for refining the segmentation is discussed. We compare our unsupervised segmentation algorithm with two popular unsupervised segmentation algorithms Mean Shift [2] and Efficient Graph-Based Image Segmentation [3], and show that our algorithm can produce better semantic segmentation results with acceptable computational overhead. Fig. 1 illustrates the process of our approach.

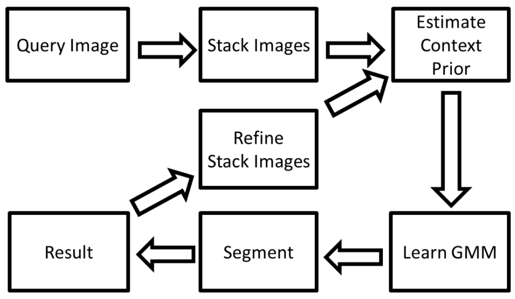

Fig. 1 The process of our approach.

3 Scene Structure Analysis

The gist descriptor [14] has been shown to be effective in modeling the global structure of the scene in an image. We may measure the similarity between the scene structures of two images by comparing their gist descriptors. Given an input image, we compute its gist descriptor, and then match each image in the database with the input image by the similarity between their gist descriptors and retrieve $N = 100$ most similar images. We collect these candidate images as an image stack: $\mathcal{I} = \{I_1, I_2, I_3, \ldots, I_N\}$. Some examples are shown in Fig. 2.

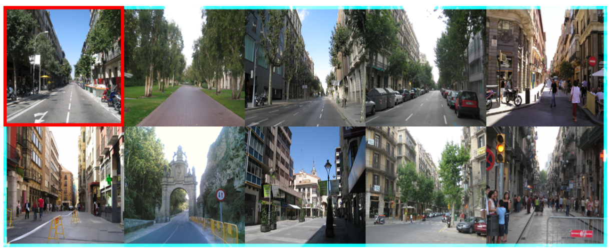

Fig. 2 An input image (top-left, surrounded by red rectangle) and the image stack retrieved by matching the gist descriptors. Here we show only 9 images of the stack.

Although the images in the image stack exhibit scene structures similar to the one of the input image, to extract a common partition that corresponds to the similar structures is not straightforward. As can be seen in Fig. 2, the appearance of a local region at the same location across different images is diverse. Moreover, neighboring pixels or patches in the same image also have varied colors or textures, and cannot be easily grouped into semantically or geometrically meaningful regions. This situation can be observed in the results produced by some standard unsupervised segmentation algorithm such as the Efficient Graph-Based Image Segmentation algorithm [3].

We propose to use the sparse coding technique for aggregating the appearance information of the image stack. Local image features are extracted using Gabor filters. For each image in the image stack, we compute a bank of Gabor filters of eight orientations over four scales for RGB channels. We choose to use eight orientations instead of the popular setting of six orientations since we would like to model subtler variations of directional features. Moreover, because large-scale features are less informative for image segmentation than for holistic image classification and the images used in our experiments are of a moderate size (256×256 pixels), we use only four scales of Gabor filters rather than five scales to model the local image features. We find that such a setting is empirically suitable for our application. The Gabor filters used in this work are shown in Fig. 3. The filter responses at each pixel hence yield a 96-dimensional vector representing local textures around that pixel, since $96 = 8 \times 4 \times 3$.

4 Estimating the Contextual Prior Using Sparse Coding

4.1 Off-line Training for Sparse Dictionary

We use the sparse coding technique and the toolbox developed by [12] to train a sparse coding dictionary

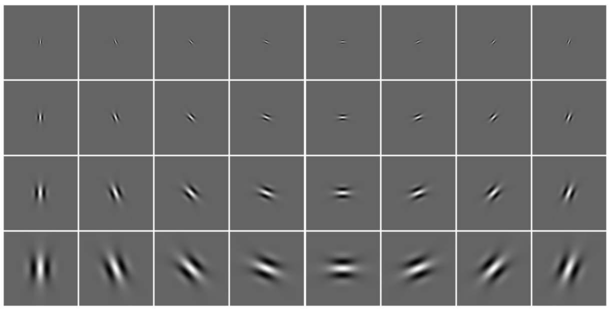

Fig. 3 The Gabor filters used in our task: eight orientations over four scales.

D of visual words from the Gabor filters responses. We extract $L = 10{,}000$ Gabor filter response vectors $\{\mathbf{x}_l | \mathbf{x}_l \in \mathbb{R}^m\}_{l=1}^L$ from several scene images, which are randomly selected from the LabelMe database [22]. As mentioned earlier, responses of the Gabor filter bank yield a 96-dimensional vector, and therefore we have $m = 96$. Given $\{\mathbf{x}_l\}_{l=1}^L$, the goal is to train dictionary **D**, which can be expressed as an m-by-k matrix with columns $\{\mathbf{d}_j\}_{j=1}^k$, where $\mathbf{d}_j$ is an m-dimensional vector that represents a visual word. In our experiments the dictionary **D** contains $k = 200$ visual words. We find that 200 visual words are sufficient for analyzing the structures of general scene images. The dictionary **D** is obtained by solving the following optimization

$$\min_{\mathbf{D}\in\mathcal{C},\{\boldsymbol{\alpha}_l\in\mathbb{R}^k\}} \frac{1}{L}\sum_{l=1}^{L}\left(\frac{1}{2}\|\mathbf{x}_l - \mathbf{D}\boldsymbol{\alpha}_l\|_2^2 + \lambda\|\boldsymbol{\alpha}_l\|_1\right), \tag{1}$$

where $\mathcal{C}$ is a convex set of matrices satisfying the constraint

$$\mathcal{C} \triangleq \{\mathbf{D} \in \mathbb{R}^{m\times k} \text{ s.t. } \forall j = 1, \ldots, k,\ \mathbf{d}_j^T\mathbf{d}_j \leq 1\}, \tag{2}$$

and the ℓ_1 penalty is added to enforce sparse decomposition coefficients $\{\boldsymbol{\alpha}_l | \boldsymbol{\alpha}_l \in \mathbb{R}^k\}_{l=1}^L$. The value of parameter λ is 0.15 in our experiments. More detailed information about training sparse dictionary can be referred to [12]. Note that we only need to perform off-line training once to learn a sparse dictionary. We may use the same sparse dictionary for all input images subsequently.

Some recent works on sparse coding suggest that sparse dictionaries consisting of local bases can help to achieve better performance in classification tasks, *e.g.* [28] and [29]. In particular, Yang *et al.* [28] present a sparse coding method that uses a mixture model of over-complete dictionaries. Such a sparse coding scheme may be taken into consideration for improving the descriptiveness of dictionaries. Nevertheless, in this work we simply use a single dictionary to derive sparse coding. Instead of learning a mixture of local dictionaries, we resort to the statistics of multiple observations given by an image stack for exploiting the local property of the sparse coding subspaces. The details of extracting the cluster information from sparse coding are described in the next section.

4.2 Contextual Prior Estimation

With the learned dictionary, the Gabor filter responses of each pixel can be represented as a sparse combination of a few visual words in the dictionary. Given a new input image, we may build its image stack as described in section 3. For each image I_n in the image stack $\mathcal{I}$ of N images, we compute the sparse coefficients $\boldsymbol{\beta}_i$ at each pixel position i by solving

$$\min_{\boldsymbol{\beta}_i\in\mathbb{R}^k} \frac{1}{2}\|\mathbf{x}_i - \mathbf{D}\boldsymbol{\beta}_i\|_2^2 + \lambda\|\boldsymbol{\beta}_i\|_1, \tag{3}$$

where $\mathbf{x}_i$ is the Gabor-filter response vector at pixel i. Let $\beta_{i,j}(n)$ denote the coefficient of the jth visual word at pixel i of the nth image in the image stack. At each pixel position i we compute a k-dimensional vector $\mathbf{v}_i$ to record the (signed) occurrence of each visual word. The value of the jth element of $\mathbf{v}_i$ is given by

$$v_{i,j} = \sum_{n=1}^{N} \operatorname{sign}(\beta_{i,j}(n)), \quad \text{for } j = 1, \ldots, k, \tag{4}$$

where $k = 200$ as previously mentioned. Thus, $\mathbf{v}_i$ aggregates the signed counts of the occurrences of visual words at position i across all N images in the image stack $\mathcal{I}$.

If a visual word has similar responses of sparse coefficients across all images in the image stack, the signed occurrence would have a large magnitude and it means that this visual word is more reliable and descriptive. On the other hand, if a visual word has diverse or random responses of sparse coefficients across all images, the terms of the aggregated signed counts in (4) are more likely to cancel out each other, and such a visual word should be less significant. Furthermore, by the signed counts we assume that the coefficients across all images have equal contributions, and thus the rationale of using the counts rather than the original values is to prevent being dominated by a few outliers of large absolute values.

We expect that if two visual-word occurrence vectors $\mathbf{v}_i$ and $\mathbf{v}_j$ at the pixel positions i and j are similar, the two pixel positions probably correspond to the same region. Such kinds of spatial and semantic relations can be obtained *a priori* from the image stack without looking at the specific content of the input image to be segmented. We then use k-means clustering to group all $\{\mathbf{v}_i\}$ for all pixel positions, and each pixel

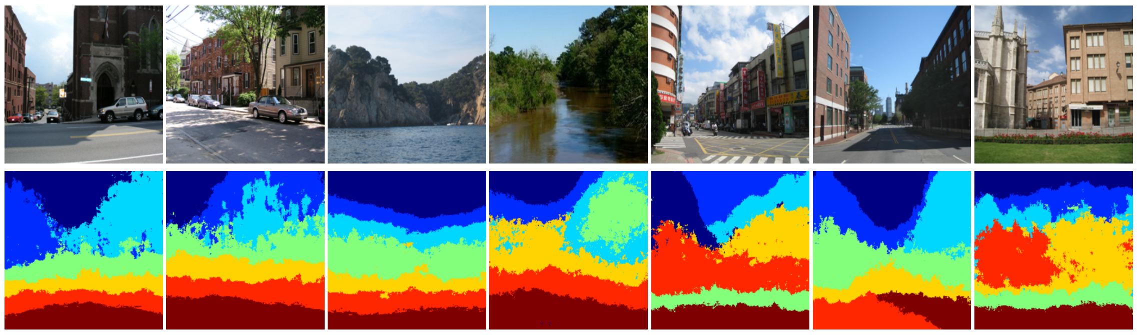

Fig. 4 Top: Input images. Bottom: The contextual priors, *i.e.*, the k-means clusters derived from the sparse coding dictionary and the image stacks.

position i will be associated with one of the clusters. We call the clustering result as the *contextual prior*, which provides the prior information of how the pixel positions should be grouped. Fig. 4 shows some examples of input images and their contextual priors. Compared with the input images we can observe that the semantic regions of the input images are roughly aligned with the k-means clusters. In our experiments, we set the number of clusters to be seven for k-means clustering, and, in general, k-means clustering is able to yield reliable results.

5 Markov-Random-Field (MRF) Based Image Segmentation

Given an input image to be segmented, we may compute its contextual prior using the method described in the previous section. We will incorporate the contextual prior information into the segmentation process, and the image segmentation will be achieved by solving an MRF-based optimization. To begin with, we build a Gaussian mixture model (GMM) for each k-means cluster, with respect to the contextual prior. The idea of modeling the region properties by GMMs has been proposed in GrabCut [17]. Note that GrabCut requires the user to provide an initial bounding box around the foreground object. In [17] the GMMs for the initial foreground and background regions can then be built and iteratively updated to produce the final segmentation. In our work, we use the contextual prior to replace the hint provided by the user, and our algorithm partitions the input image into multiple regions rather than resolving figure/ground segmentation. We use the gmdistribution class in the Matlab Statistics toolbox to learn the GMMs of pixels' RGB values according to the contextual priors. Each GMM has five components, and the regularization parameter is 0.001.

With the GMMs being constructed for all contextual prior clusters, our unsupervised segmentation algorithm solves the following MRF-based energy minimization problem to find an optimal label assignment Γ to all pixels in the input image:

$$\arg\min_{\gamma_i \in \Gamma} \sum_i \Big\{ \phi(\mathbf{z}_i, \tau_i, \gamma_i) + \eta \sum_{j \in \Omega(i)} \psi(\mathbf{z}_i, \mathbf{x}_i, \gamma_i, \mathbf{z}_j, \mathbf{x}_j, \gamma_j) \Big\} , \tag{5}$$

where $\mathbf{z}_i$ is the vector of RGB values of pixel i, $\mathbf{x}_i$ is the vector of Gabor responses at pixel i, and τ_i is the GMM component to which the pixel i belongs. We consider a four-connected neighborhood $\Omega(i)$ of i for the smoothness term, and $\gamma_i \in \Gamma$ is the label assigned to pixel i. The parameter η $(= 100)$ is to balance the weights between different terms. In our experiments the value of τ_i can be $\{1, \ldots, 5\}$ since we set a GMM to have five components (*i.e.*, a mixture of five Gaussians). Label γ_i takes a value from $\{1, \ldots, 7\}$ since we partition an image into at most seven regions. For most of the images in the dataset, we notice that the number of different semantic regions included in a single image is rarely greater than seven.

The first term in (5), *i.e.* the data term, is defined by

$$\begin{aligned} &\phi(\mathbf{z}_i, \tau_i, \gamma_i) \\ &= -\log p(\mathbf{z}_i | \tau_i, \gamma_i) - \log \pi_{\tau_i,\gamma_i} \\ &= \frac{1}{2} \log \det \boldsymbol{\Sigma}_{\tau_i,\gamma_i} + \frac{1}{2} (\mathbf{z}_i - \boldsymbol{\mu}_{\tau_i,\gamma_i})^T \boldsymbol{\Sigma}^{-1}_{\tau_i,\gamma_i} (\mathbf{z}_i - \boldsymbol{\mu}_{\tau_i,\gamma_i}) \\ &\quad - \log \pi_{\tau_i,\gamma_i} + \text{const.} \end{aligned} \tag{6}$$

The mean and covariance of Gaussian component corresponding to component τ_i and label γ_i are denoted by $\boldsymbol{\mu}_{\tau_i,\gamma_i}$ and $\boldsymbol{\Sigma}_{\tau_i,\gamma_i}$, and π_{τ_i,γ_i} are the mixture weights indexed by the values of τ_i and γ_i. The second term in (5) is a smoothness term that enforces consistent labeling of neighboring pixels:

$$\begin{aligned}&\psi(\mathbf{z}_i, \mathbf{x}_i, \gamma_i, \mathbf{z}_j, \mathbf{x}_j, \gamma_j)\\ &= (1 - \delta(\gamma_i - \gamma_j)) \Big\{ \exp(-\theta_{\text{Color}}\, ||\mathbf{z}_i - \mathbf{z}_j||^2)\\ &\qquad + \kappa \exp(-\theta_{\text{Gabor}}\, ||\mathbf{x}_i - \mathbf{x}_j||^2) \Big\}, \end{aligned} \tag{7}$$

where $\delta(\cdot)$ is the Kronecker delta and a weight $\kappa = 0.2$ is used in all experiments. The scale factors θ_{Color} and θ_{Gabor} are given by

$$\theta_{\text{Color}} = (2\langle||\mathbf{z}_i - \mathbf{z}_j||^2\rangle)^{-1} \tag{8}$$

and

$$\theta_{\text{Gabor}} = (2\langle||\mathbf{x}_i - \mathbf{x}_j||^2\rangle)^{-1}, \tag{9}$$

where $\langle\cdot\rangle$ denotes the exception value over an image.

The GMM-MRF energy minimization problem in (5) can be solved by the graph-cuts software GCMEX [1] in two seconds for a 256-by-256 color image. After we solve the energy minimization, the final number of remaining regions might decrease and we might obtain fewer than seven regions, depending on the content of the input image.

6 Refining the Image Stack

Through experiments we observe that some images in the image stack retrieved by comparing gist descriptors cannot be aligned with the input image very well. Including these images in stack might deteriorate the effectiveness of the estimated contextual prior. We may use the segmentation result produced by our algorithm to refine the image stack, and then run our algorithm again to obtain a new segmentation result. We find that the segmentation often can be improved after this refinement step. The refinement is done by removing unsuitable images from the image stack. The quality of each image in the image stack should be evaluated by the agreement between its edges and the region boundaries suggested by the preliminary segmentation result. More specifically, we use the *bidirectional chamfer distance* [8] as the metric to compare the boundaries of regions in the preliminary segmentation with the edges of each image in the image stack. We keep only the top 20% most consistent images to form a refined image stack, and then re-run the algorithm to obtain a new segmentation.

7 Experiments

We use the LabelMe database [22] and the SUN database [27] to test our segmentation algorithm. From the two databases we pick 78,321 images that consist of street views, urban landscapes, and natural scenes. We use these images to build a dataset for our experiments. Each image in our dataset is resized to 256-by-256 pixels. We use the dataset to collect image stacks and to generate contextual priors. Furthermore, the LabelMe database provides label information, which can later be used as ground truth for evaluation. Fig. 5 shows some results produced by our algorithm. Each row in Fig. 5 contains an input image, its contextual prior, and the preliminary segmentation, followed by the new contextual prior derived from the refined image stack, and the refined segmentation. As can be seen in Fig. 5, most of the boundaries and regions are aligned well with the semantic scene structures. When the segmentation results generated by the first-round segmentations are moderately reliable, the refinement step is usually able to produce more stable segmentation results. It can also be observed that the contextual priors become more specific at the refinement step, *c.f.* Fig. 5(b) versus Fig. 5(e). The whole process of producing the segmentation results is unsupervised; it is done automatically without using the label information provided by the LabelMe database. The parameters in our algorithm are fixed during the experiment; their values are chosen as described in the previous sections.

7.1 Computational Issues

We have precomputed the gist descriptors and the sparse coefficients $\boldsymbol{\beta}_i$ of the images in our dataset, and have stored them with the respective images. Note that we do not have to store the intermediate 96-dimensional vectors of the Gabor filter responses. After we use the Gabor filter responses to decide the sparse coefficients, we may ignore the Gabor filter responses and store only the sparse coefficients. Each image requires 3MB space to store its sparse coefficients, and therefore totally it needs 235GB space to store all of the sparse coefficients for the 78,321 images in our dataset. Given an input image, we may directly compare its gist descriptors with the stored ones, and on-line build its image stack of 100 images that exhibit similar scenes. We may run through the 100 images one by one, and hence we do not need to load the sparse coefficients of all 100 images at once. The visual-word occurrence vectors $\mathbf{v}_i$ in (4) can thus be computed incrementally from the stored coefficients $\boldsymbol{\beta}_i$. This way the memory consumption can be well handled even if we use an image stack of a larger size. The typical run-time for each step in our algorithm is as follows: running k-means for the contextual priors takes 3 seconds, learning GMMs takes 4 seconds, and performing graph-cuts needs 2 seconds. The experiments are done in MATLAB environment, on a 4-core 2.8GHz CPU with 12GB memory. Note that if we do not precompute the gist descriptors and the sparse coefficients, the run-time will be dominated by these computations, which requires about 400 seconds.

7.2 Comparisons

We compare our results with the segmentations produced by Mean Shift [2], the Efficient Graph-Based Image Segmentation [3], and the method of Russell *et al.* [19]. Fig. 6 shows some example segmentations of different input images. We have manually adjusted the parameters for Mean Shift and Graph-Based Segmentation such that they can generate a suitable number of visually plausible regions that mostly resemble the semantic scene structures. Both of Mean Shift and Graph-Based Segmentation are very efficient; the segmentation of an input image can be generated in one second. For the comparison with the method of [19], instead of using 5,000 images to build the image stack as in the original setting of [19], we test their algorithm using an image stack of only 100 images to see if the algorithm can still produce reliable segmentation results. The typical run-time of the method of [19] with the modified setting is 1,100 seconds.

As shown in Fig. 6, the segmentations produced by the Graph-Based Segmentation are somewhat not stable, and the segmented regions do not reflect the semantic scene structures very well. The segmentations generated by Mean Shift seem to be more stable and detailed, and the region boundaries better fit the strong edges. Nevertheless, the segmented regions found by Mean Shift still do not correspond to the semantic scene structures very well. The regions might look plausible due to the similarity in color, but they reflect less information about the context of the scene. The regions produced by the method of [19] using an image stack of 100 images can roughly reflect the semantics and the contexts in the scenes, but the overall segmentation quality is not good enough. In Fig. 7 we show the heat maps of scene components estimated by the method of [19]. The heat maps are used to generate the data term of the energy minimization in the method of [19], and it can be seen that the segmentation result greatly relies on the quality of the estimated scene components. Using a stack of only 100 images might not be enough for the method of [19] to yield good segmentation results. In contrast, our method can more effectively derive the context information from a stack of 100 image using the sparse coding technique. Moreover, the contextual priors are simply used as the initial hint in our GMM-MRF energy minimization scheme, and the segmentation produced by our algorithm is more flexible to adapt to the specific content of the input image.

To further evaluate the qualities of the segmentations produced by different algorithms, we employ the human-labeled segmentations as the references for comparing different segmentation results. The LabelMe database includes human-annotated labels on each image, and therefore we may use these labels to generate ground-truth segmentations. We consider the following categories of semantic regions: building, cliff, foliage, ground, mountain, road, sea, sidewalk, sky, tree, and water. Labels of other categories are ignored and considered as don't care when we evaluate the segmentation performance. From the LabelMe database we take 40 images that present a variety of scenes, and generate the ground-truth segmentation for each image.

We use the Adjusted Rand Index [7] as a measure to evaluate the similarity between a segmentation result and the ground-truth segmentation. The Adjusted Rand Index is a normalized version of the Rand Index [16]. The idea of Rand Index is to check the region consistency of each pair of pixels $\{i, j\}$. Consider a result of segmentation $\mathcal{R} = \{R_1, \ldots, R_s, \ldots\}$ and the ground-truth segmentation $\hat{\mathcal{R}} = \{\hat{R}_1, \ldots, \hat{R}_t, \ldots\}$, where R_s and $\hat{R}_t$ are regions consisting of pixels. There are four possible cases for the region assignments on $\{i, j\}$ with respect to the two segmentations $\mathcal{R}, \hat{\mathcal{R}}$, and the Rand Index takes account of the following two cases:

$$\begin{aligned} &a_{ij} : i, j \in R_s \text{ and } i, j \in \hat{R}_t \,; \\ &b_{ij} : i \in R_s, j \in R_{s'} \,, \text{ and } i \in \hat{R}_t, j \in \hat{R}_{t'} \,, \\ &\qquad \text{where } s \neq s', t \neq t' \,. \end{aligned} \tag{10}$$

The Rand Index counts the number of these two cases over all possible pairs:

$$\text{Rand Index} = \frac{1}{\binom{M}{2}} \sum_{i,j} \mathbf{1}_{a_{ij}}(i, j) + \mathbf{1}_{b_{ij}}(i, j) \,, \tag{11}$$

where $\mathbf{1}_{a_{ij}}$ and $\mathbf{1}_{b_{ij}}$ are the indicator functions and M is the number of pixels in the image. The value is maximized if two segmentations are identical. Note that the Rand Index does not have fix values for random segmentations. To resolve the problem, the Adjusted Rand Index normalizes the Rand Index to 0 for random segmentations. The maximum value of the Adjusted Rand Index is 1 also due to the normalization. The Adjusted Rand Index is defined as the ratio of 'the Rand Index subtracting the expected Rand Index' to 'the maximum Rand Index subtracting the expected Rand Index'.

Fig. 8 illustrates the evaluations of comparing the segmentations with the ground truths using the Adjusted Rand Index. The refined segmentations produced by our algorithm are most consistent with the ground truths according to the Adjusted Rand Index scores. We sort the images by the Adjusted Rand Index scores of the refined segmentations, and arrange the images accordingly in descending order for visualization. The scores of the refined segmentations roughly form an envelope of the scores of other methods, which means that our algorithm is better at generating semantically meaningful segmentations in most cases. Furthermore, the average score of our refined segmentations is 0.8010, higher than the average score of Mean Shift (0.7611), the Graph-Based Segmentation (0.7411), and the method of [19] (0.7388). The experimental results suggest that, even though our algorithm only uses a small number of additional images as the referenced image stack, our algorithm is effective enough to improve scene segmentation and obtain semantic regions via exploiting the contextual prior derived from the sparse coding representation of the image stack.

7.2.1 Number of Visual Words

In the aforementioned experiment, the sparse coding dictionary **D** of our method contains 200 visual words. We have also tried to build dictionaries of two different sizes: 100 visual words and 400 visual words. We repeat the experiment by changing the number of visual words while keeping the remaining parameters and settings unchanged. The segmentation results are evaluated as in the previous experiment, and the performances of using different numbers of visual words are shown in Fig. 9. In general, the performances do not vary a lot, except that in a few cases the performance of 100 or 400 visual words drops drastically. The abnormal performance of 400 visual words might be due to the curse of dimensionality: The refined image stack contains only 20 images. The dimensionality of the 400-visual-word vector may be too high such that it is insufficient to obtain reliable clustering results from only 20 images.

7.2.2 Image Stack Size

In Fig. 10 we evaluate the performances of our method and the method of Russell *et al.* [19] using different numbers of images to form the image stack. We have tested three different settings, using 100 images, 500 images, and 1,000 images in the image stack. As in the previous experiments, the segmentation results are compared with the ground truth to compute the Adjusted Rand Index scores. During the refinement step, we keep the top 20% most consistent images to form a refined image stack. Therefore, the refined image stack would contain 20, 100, and 200 images with respect to the number of 100, 500, and 1,000 images in the initial image stack. For our method, the results of using 500 or 1,000 images in the image stack are not better than the result of using 100 images. It might suggest that to include too many images in the image stack would probably deteriorate the effectiveness of the estimated contextual prior, since it is not easy to find 500 or 1,000 images that are all well aligned with the input image. We find that using an image stack of 100 images is suitable for our application.

8 Conclusion

We have presented an unsupervised segmentation algorithm for partitioning an image into semantically meaningful regions. Our algorithm employs the sparse coding technique to estimate the contextual prior for an input image. The contextual prior is able to integrate the spatial information and the sparse representation of the scene. We use the contextual prior as the guidance to construct Gaussian mixture models for different parts of the input image. The Gaussian mixture models are incorporated into an MRF-based energy-minimization problem that can be solved efficiently by graph-cuts. We believe that the idea of combining sparse representation with context information should be worth further investigation for more applications.

We have shown that the information of contextual prior learned from the image stack is useful in localizing the semantic regions of input image. A possible extension of the current algorithm is to include more complex local features into the GMM when constructing the MRF-based minimization. The additional information provided by local features should be able to improve the reliability of graph-cuts segmentation. Such information can also be applied to a subsequent classification-based algorithm for region labeling, which will be helpful in making progress towards the goal of scene parsing and image understanding.

Acknowledgements For the final version, please refer to "Unsupervised Scene Segmentation Using Sparse Coding Context", Machine Vision and Applications, February 2013, Volume 24, Issue 2, pp 243-254.

References

1. Boykov, Y., Veksler, O., Zabih, R.: Fast approximate energy minimization via graph cuts. IEEE Trans. Pattern Anal. Mach. Intell. **23**(11), 1222–1239 (2001)

2. Comaniciu, D., Meer, P.: Mean shift: A robust approach toward feature space analysis. IEEE Trans. Pattern Anal. Mach. Intell. **24**(5), 603–619 (2002)
3. Felzenszwalb, P.F., Huttenlocher, D.P.: Efficient graph-based image segmentation. International Journal of Computer Vision **59**(2), 167–181 (2004)
4. Hays, J., Efros, A.A.: Scene completion using millions of photographs. Commun. ACM **51**(10), 87–94 (2008)
5. Hoiem, D., Efros, A.A., Hebert, M.: Geometric context from a single image. In: ICCV, pp. 654–661 (2005)
6. Hoiem, D., Efros, A.A., Hebert, M.: Closing the loop in scene interpretation. In: CVPR (2008)
7. Hubert, L., Arabie, P.: Comparing partitions. Journal of Classification **2**(1), 193–218 (1985)
8. Huttenlocher, D.P., Klanderman, G.A., Rucklidge, W.: Comparing images using the hausdorff distance. IEEE Trans. Pattern Anal. Mach. Intell. **15**(9), 850–863 (1993)
9. Joulin, A., Bach, F.R., Ponce, J.: Discriminative clustering for image co-segmentation. In: CVPR, pp. 1943–1950 (2010)
10. Liu, C., Yuen, J., Torralba, A.B.: Nonparametric scene parsing: Label transfer via dense scene alignment. In: CVPR, pp. 1972–1979 (2009)
11. Lowe, D.G.: Distinctive image features from scale-invariant keypoints. International Journal of Computer Vision **60**(2), 91–110 (2004)
12. Mairal, J., Bach, F., Ponce, J., Sapiro, G.: Online dictionary learning for sparse coding. In: ICML, p. 87 (2009)
13. Mori, G., Ren, X., Efros, A.A., Malik, J.: Recovering human body configurations: Combining segmentation and recognition. In: CVPR (2), pp. 326–333 (2004)
14. Oliva, A., Torralba, A.B.: Modeling the shape of the scene: A holistic representation of the spatial envelope. International Journal of Computer Vision **42**(3), 145–175 (2001)
15. Quack, T., Leibe, B., Gool, L.J.V.: World-scale mining of objects and events from community photo collections. In: CIVR, pp. 47–56 (2008)
16. Rand, W.: Objective criteria for the evaluation of clustering methods. J. am. Statistical Assoc. **66**(336), 846–850 (1971)
17. Rother, C., Kolmogorov, V., Blake, A.: “grabcut”– interactive foreground extraction using iterated graph cuts. ACM Trans. Graph. **23**(3), 309–314 (2004)
18. Rother, C., Minka, T.P., Blake, A., Kolmogorov, V.: Cosegmentation of image pairs by histogram matching - incorporating a global constraint into mrfs. In: CVPR (1), pp. 993–1000 (2006)
19. Russell, B.C., Efros, A.A., Sivic, J., Freeman, W.T., Zisserman, A.: Segmenting scenes by matching image composites. In: NIPS (2009)
20. Russell, B.C., Freeman, W.T., Efros, A.A., Sivic, J., Zisserman, A.: Using multiple segmentations to discover objects and their extent in image collections. In: CVPR (2), pp. 1605–1614 (2006)
21. Russell, B.C., Torralba, A.B.: Building a database of 3d scenes from user annotations. In: CVPR, pp. 2711–2718 (2009)
22. Russell, B.C., Torralba, A.B., Murphy, K.P., Freeman, W.T.: Labelme: A database and web-based tool for image annotation. International Journal of Computer Vision **77**(1-3), 157–173 (2008)
23. Shi, J., Malik, J.: Normalized cuts and image segmentation. IEEE Trans. Pattern Anal. Mach. Intell. **22**(8), 888–905 (2000)
24. Simon, I., Seitz, S.M.: Scene segmentation using the wisdom of crowds. In: ECCV (2), pp. 541–553 (2008)
25. Snavely, N., Seitz, S.M., Szeliski, R.: Modeling the world from internet photo collections. International Journal of Computer Vision **80**(2), 189–210 (2008)
26. Tu, Z., Chen, X., Yuille, A.L., Zhu, S.C.: Image parsing: Unifying segmentation, detection and recognition. In: ICCV, pp. 18–25 (2003)
27. Xiao, J., Hays, J., Ehinger, K.A., Oliva, A., Torralba, A.: Sun database: Large-scale scene recognition from abbey to zoo. In: CVPR, pp. 3485–3492 (2010)
28. Yang, J., Yu, K., Huang, T.S.: Efficient highly over-complete sparse coding using a mixture model. In: ECCV (5), pp. 113–126 (2010)
29. Yu, K., Zhang, T., Gong, Y.: Nonlinear learning using local coordinate coding. In: NIPS (2009)

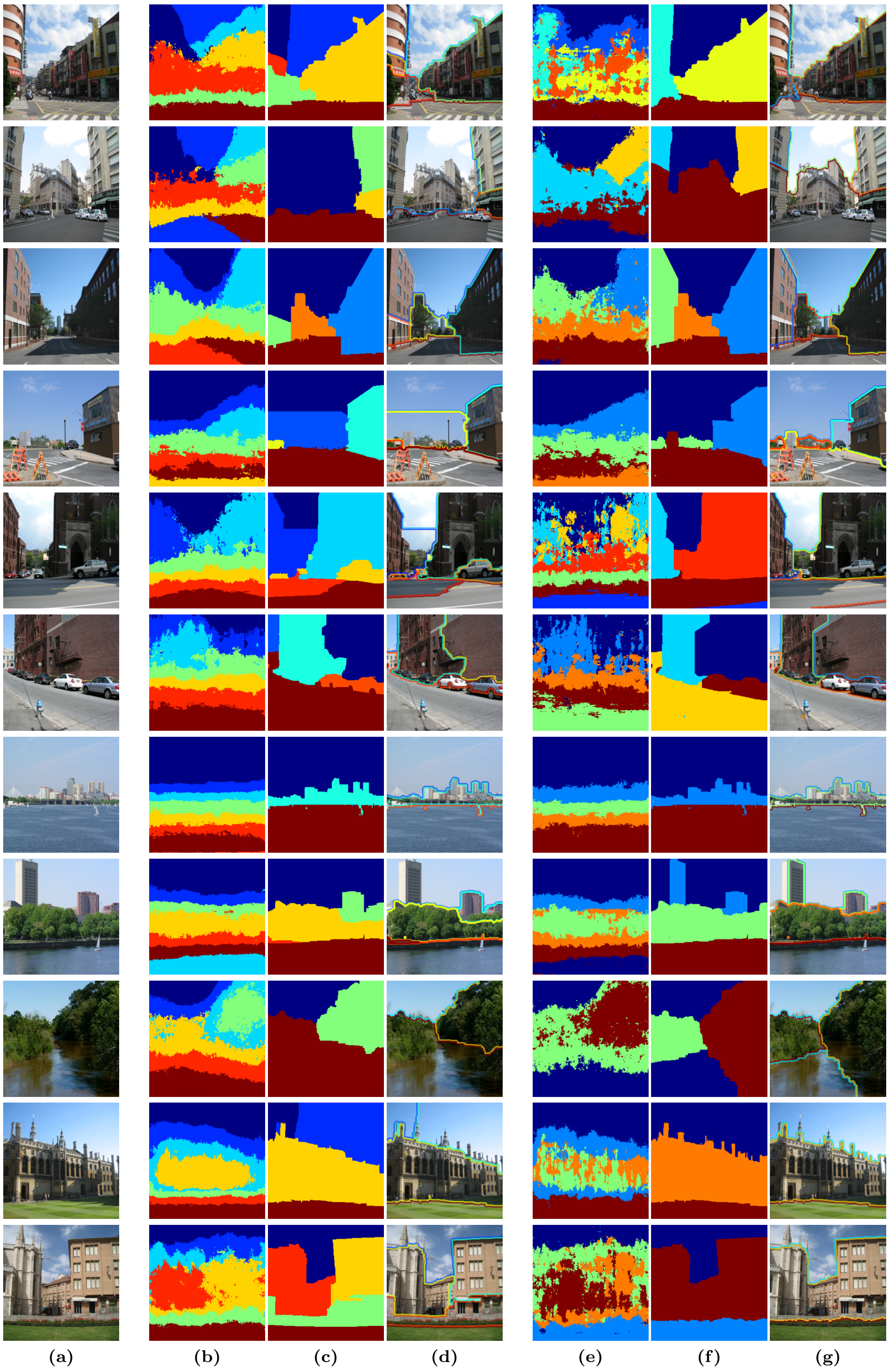

Fig. 5 Segmentation results. **(a)** Input images. **(b)** K-means contextual priors. **(c)** Preliminary segmentations. **(d)** Boundaries of (c). **(e)** Refined k-means contextual priors. **(f)** Refined segmentations. **(g)** Boundaries of (f).

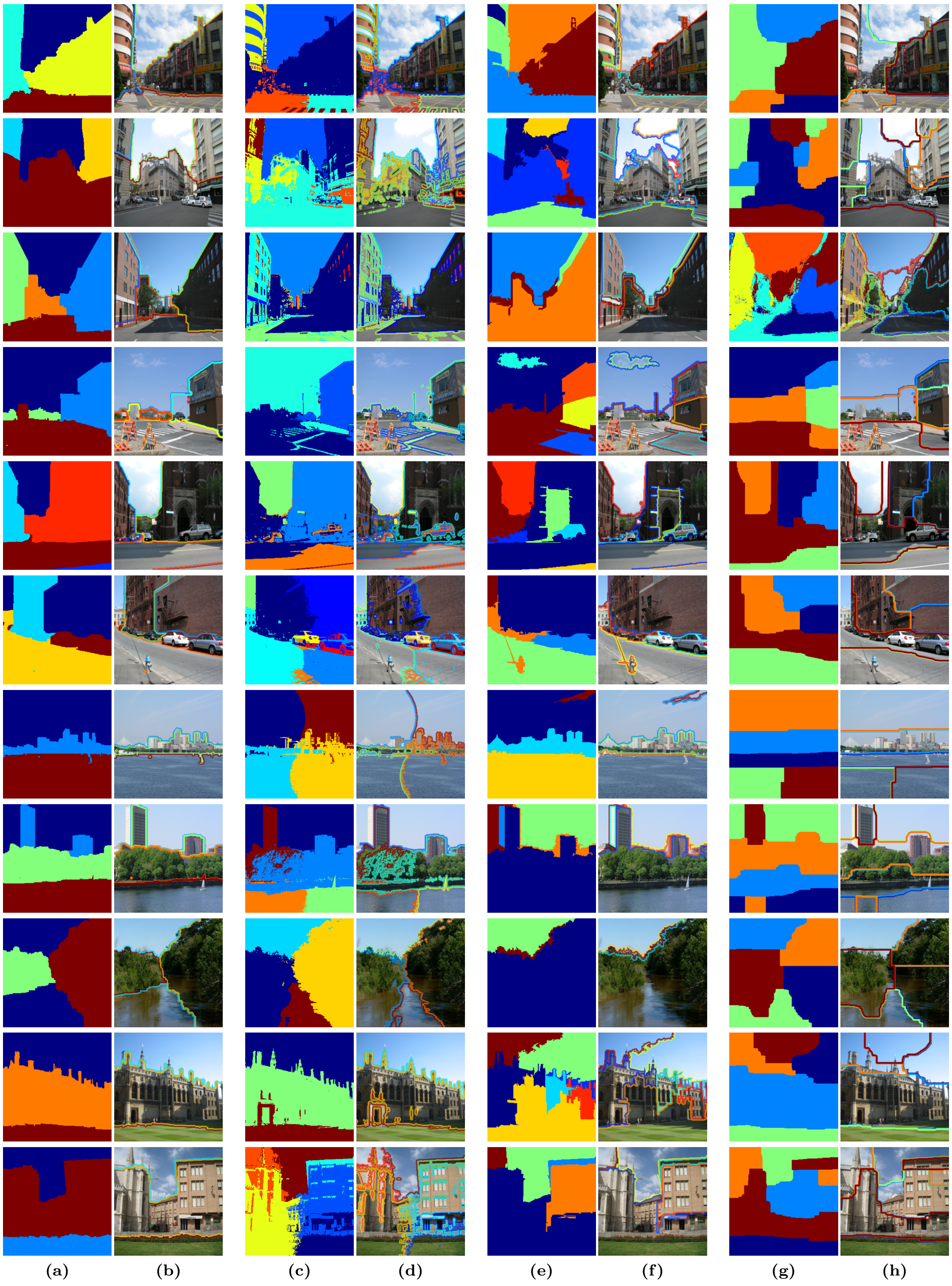

Fig. 6 Comparison. **(a)** Our results. **(b)** Boundaries of (a). **(c)** Results generated by Mean Shift [2]. **(d)** Boundaries of (c). **(e)** Results obtained by the Efficient Graph-Based Segmentation [3]. **(f)** Boundaries of (e). **(g)** Results using the algorithm of [19]. **(h)** Boundaries of (g).

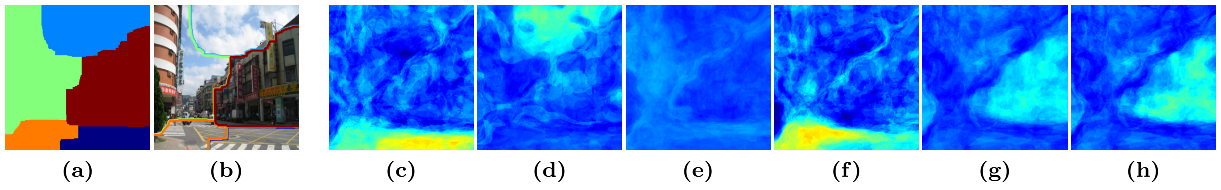

Fig. 7 **(a)** A segmentation result using the method of Russell *et al*. [19] with an image stack of 100 images. **(b)** The boundary of (a). **(c)**–**(g)**: The heat maps of scene components estimated by the method of [19]. These heat maps are used to generate the data term of the energy minimization in the method of [19]. It can be seen that the segmentation result greatly relies on the quality of the estimated scene components.

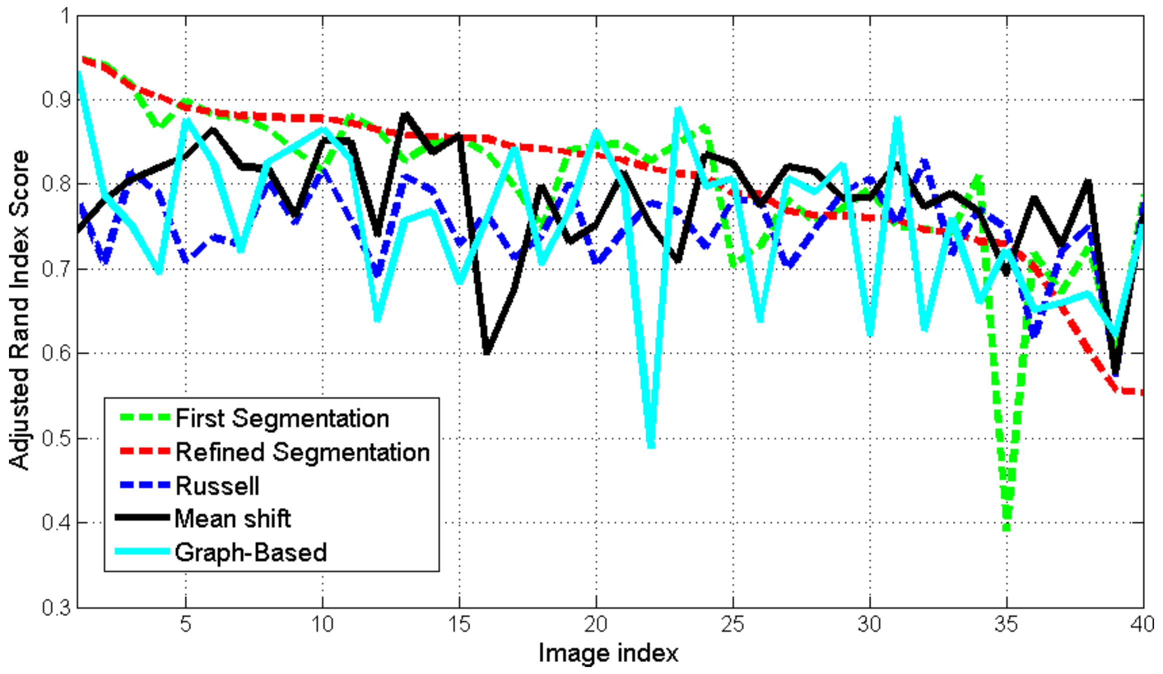

Fig. 8 Compared with the ground truth using the Adjusted Rand Index on 40 images chosen from the LabelMe database [22]. We consider the following categories of semantic regions: building, cliff, foliage, ground, mountain, road, sea, sidewalk, sky, tree, and water. Labels of other categories are ignored and considered as don't care when we evaluate the segmentation performance. Image indexes are sorted according to their Adjusted Rand Index scores of our refined segmentations. The average scores are 0.7865 (ours, first-round), 0.8010 (ours, refined), 0.7611 (Mean Shift), 0.7411 (Graph-Based), and 0.7388 (the method of Russell *et al*. [19]).

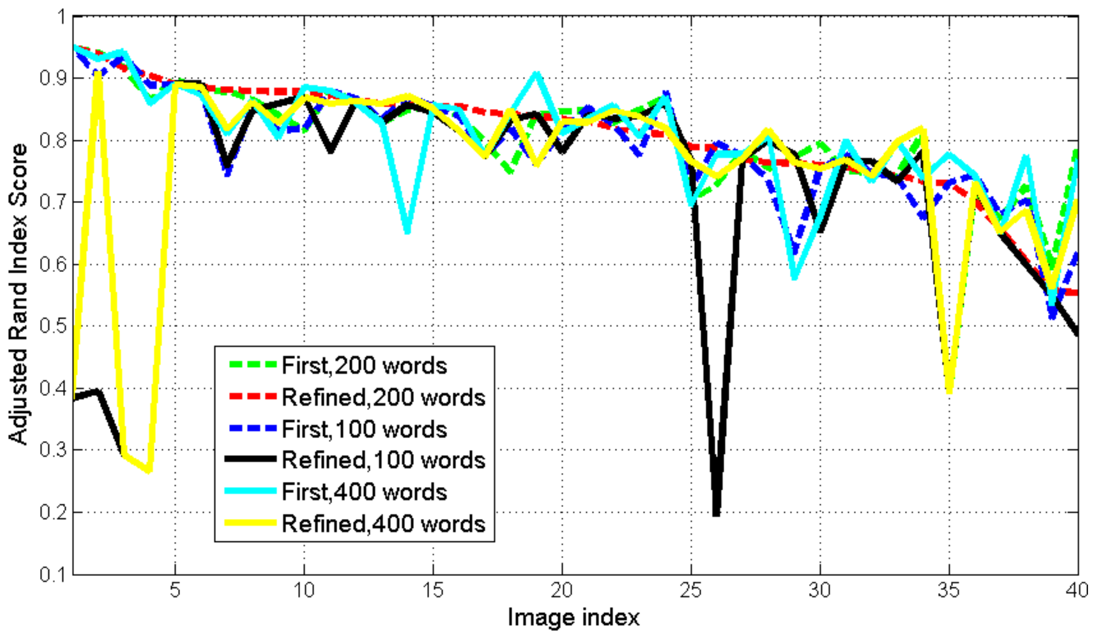

Fig. 9 A comparison on the performances of using different numbers of visual words: 100, 200, 400 visual words. For all of the three settings, the initial image stack always contains 100 images, and the refined image stack contains 20 images. The segmentation results are compared with the ground truth using the Adjusted Rand Index on the 40 images chosen from the LabelMe database as in Fig. 8. Image indexes are sorted according to their Adjusted Rand Index scores of the refined segmentations with 200 visual words. In general, the performances do not vary a lot, except in a few cases the performance of 100 or 400 visual words drops drastically.

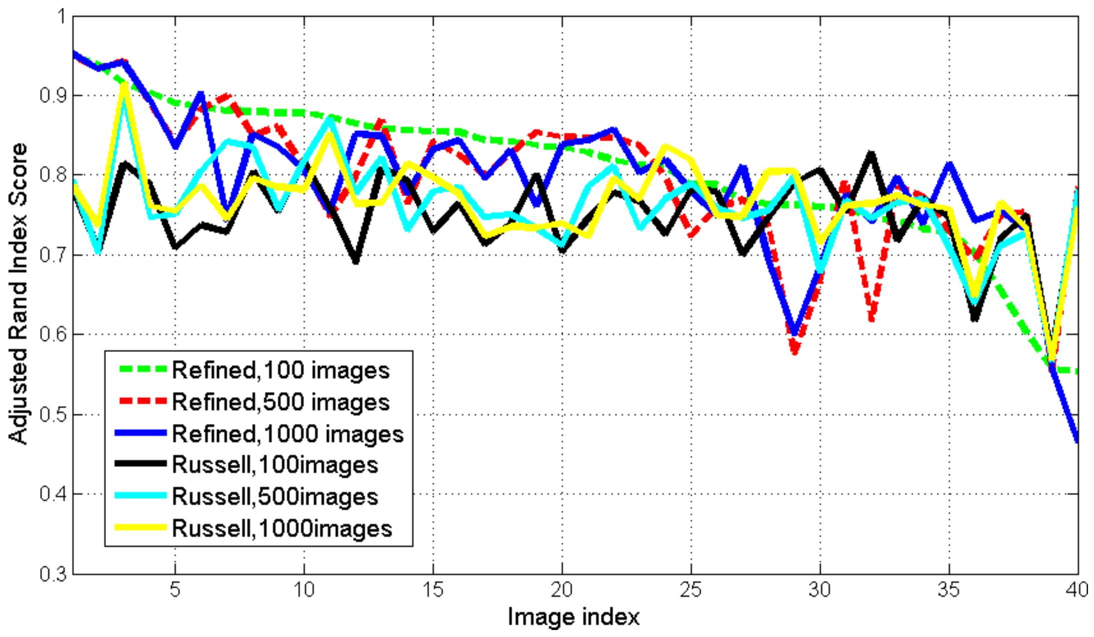

Fig. 10 The performances of our method and the method of Russell *et al.* [19] using different numbers of images in the image stack. We have tested three different settings: 100 images, 500 images, and 1,000 images. The segmentation results are compared with the ground truth using the Adjusted Rand Index on the 40 images chosen from the LabelMe database as in Fig. 8. Image indexes are sorted according to their Adjusted Rand Index scores of the refined segmentations using 100 images in the image stack. During the refinement step, we keep the top 20% most consistent images to form a refined image stack. Therefore, the refined image stack would contain 20, 100, and 200 images with respect to the initial number of 100, 500, and 1,000 images. For our method, the results of using 500 or 1,000 images in the image stack are not better than the result of using 100 images. It might imply that including too many images in the image stack would probably deteriorate the effectiveness of the estimated contextual prior.