Available online at www.sciencedirect.com
SCIENCE DIRECT®
Computer Vision and Image Understanding 95 (2004) 72–85
Computer Vision and Image Understanding
www.elsevier.com/locate/cviu

On combining graph-partitioning with non-parametric clustering for image segmentation

Aleix M. Martínez,[a,*] Pradit Mittrapiyanuruk,[b] and Avinash C. Kak[b]

[a] *Department of Electrical and Computer Engineering, The Ohio State University, OH 43210, USA*
[b] *School of Electrical and Computer Engineering, Purdue University, West Lafayette, IN 47907, USA*

Received 10 June 2002; accepted 30 January 2004
Available online 8 May 2004

Abstract

The goal of this communication is to suggest an alternative implementation of the k-way Ncut approach for image segmentation. We believe that our implementation alleviates a problem associated with the Ncut algorithm for some types of images: its tendency to partition regions that are nearly uniform with respect to the segmentation parameter. Previous implementations have used the k-means algorithm to cluster the data in the eigenspace of the affinity matrix. In the k-means based implementations, the number of clusters is estimated by minimizing a function that represents the quality of the results produced by each possible value of k. Our proposed approach uses the clustering algorithm of Koontz and Fukunaga in which k is automatically selected as clusters are formed (in a single iteration). We show comparison results obtained with the two different approaches to non-parametric clustering. The Ncut generated oversegmentations are further suppressed by a grouping stage—also Ncut based—in our implementation. The affinity matrix for the grouping stage uses similarity based on the mean values of the segments.

* Corresponding author. 1-765-494-0880.
E-mail addresses: aleix@ecn.purdue.edu (A.M. Martínez), mitrapiy@ecn.purdue.edu (P. Mittrapiyanuruk), kak@ecn.purdue.edu (A.C. Kak).

1077-3142/$ - see front matter © 2004 Elsevier Inc. All rights reserved.
doi:10.1016/j.cviu.2004.01.003

1. Introduction

Image segmentation is an important first step in much of computer vision. Several algorithms have been introduced to tackle this problem. Among them are approaches based on graph partitioning [9,19,21,22,26]. The graph approaches carry the appeal of strong theoretical basis and the advantage of being applicable not only to the segmentation of images, but also to other low, mid, and high level vision tasks dealing with mid-level grouping and model-fitting [9,17,18].

For grouping pixels into regions with a graph-theoretic approach, a graph is usually defined as $G = (V, E)$, where the nodes V represent the pixels (one node per pixel) and the edges E represent the weights $w(i, j)$ that connect pairs of nodes. E is generally represented by an $n \times n$ matrix, where n is the number of pixels in the image. One of the most frequently used techniques to partition a graph is by means of the *cut* cost function [1,26]. The goal of the cut algorithm is to find two sub-graphs A and B of G that minimize the value of

$$\mathrm{cut}(A, B) = \sum_{i \in A, j \in B} w(i, j) \tag{1}$$

and with the obvious constraints $A \cup B = V$, $A \cap B = \emptyset$, and $A \neq \emptyset$, $B \neq \emptyset$. Several alternatives to the above criterion have been proposed to date [4,5,11,20,21,24,25]. Of particular note is the normalized cut criterion (Ncut) of Shi and Malik [21], which attempts to rectify the tendency of the *cut* algorithm to prefer isolated nodes of the graph (as shown in Fig. 1A). The Ncut criterion consists of minimizing

$$\mathrm{Ncut}(A, B) = \frac{\mathrm{cut}(A, B)}{\mathrm{assoc}(A, V)} + \frac{\mathrm{cut}(A, B)}{\mathrm{assoc}(B, V)}, \tag{2}$$

where $\mathrm{assoc}(A, V) = \sum_{i \in A, j \in V} w(i, j)$, which intuitively represents the connection cost from the nodes in the sub-graph A to all nodes in the graph V.

By dividing the graph into two disjoint parts as given by the eigenvector corresponding to the second smallest eigenvalue[1] of the Laplacian [7,12], we obtain a *2-way* partition of the graph. In most cases, however, we usually want to partition (segment) an image into a larger number of parts; i.e., we want a k-way partitioning algorithm which divides our image into k parts. We can achieve this hierarchically by dividing each resulting sub-graph into two other disjoint groups until no further division is necessary (which will happen when the vertices of that subgroup are similar enough to each other) [1,21,26].

The method described in the preceding paragraph although adequate is time consuming because we need to apply our algorithm at each new iteration of the hierar-

[1] The smallest eigenvalue of the Laplacian matrix is always zero and its associated smallest eigenvector is all ones. The magnitude of the second smallest eigenvalue is related to the "fullness" of the connections in the graph (in the sense of the nodes being connected with large values of $w(i, j)$). With regard to the interpretation to be given to the corresponding eigenvector, note that each element of the eigenvector stands for a pixel location in the image. The ith element of this eigenvector tells us how much the ith pixel in the image is connected with the rest of the image [7,15].

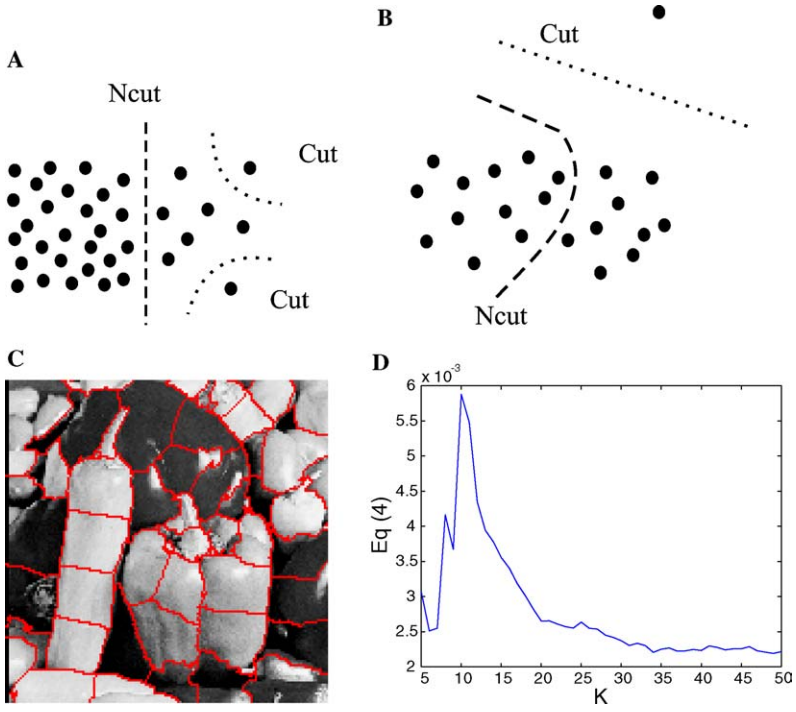

Fig. 1. (A) Cut tends to prefer isolated vectors (adapted from [21]). (B) Ncut can produce more-or-less balanced partitions. (C) An example of Ncut producing partitions at areas with constant brightness. (D) Graphical representation of Eq. (4); for which the minimum is 48—the corresponding result is shown in (C).

chy. Ideally, we would like to have a direct k-way algorithm which outputs the k disjoint areas in a single iteration [3,6,10]. A common solution is to make use of more than one single eigenvector for classification [1,12]. By using e eigenvectors starting from the second smallest, we can convert our partitioning problem into a clustering problem. Intuitively, while the second smallest eigenvector divides the graph into two parts, consecutive eigenvectors will add extra possible partitions (i.e., we will obtain more detailed segmentations as the number of eigenvectors increases). Indeed, it has been proven for the cut algorithm [2] that the more eigenvectors one uses, the better the results are (in the sense of finer results).[2]

Shi and Malik [21] define a new criterion that can be used in a k-way algorithm,

$$\mathrm{Ncut}_k(A_1, A_2, \ldots, A_k) = \frac{\mathrm{cut}(A_1, V - A_1)}{\mathrm{assoc}(A_1, V)} + \frac{\mathrm{cut}(A_2, V - A_2)}{\mathrm{assoc}(A_2, V)} + \cdots + \frac{\mathrm{cut}(A_k, V - A_k)}{\mathrm{assoc}(A_k, V)}, \tag{3}$$

[2] Intuitively, one can view the eigenvector decomposition as an approximation of the original space of G. The more eigenvectors we add to our eigenspace, the closer we will get to the original representation.

where A_i is the ith sub-graph of G. Tal and Malik [23] used the k-means algorithm to find a pre-selected number of clusters within the space spanned by the non-zero, smallest e eigenvectors. For those cases where the number of clusters is not known, the authors proposed using several values of k and then selecting that k which minimized the criterion

$$\mathrm{Ncut}_k(A_1,\ldots,A_k)/k^2. \tag{4}$$

While the Ncut criterion alleviates the tendency of the cut algorithm to isolate individual nodes if they are "distant" from the rest, it appears to have its own shortcomings, some of them we believe caused by the structure of the Ncut criterion and some by issues related to how this criterion would generally be implemented.

There is the tendency of the Ncut criterion to fragment image areas that are nearly homogeneous with respect to the segmentation parameter, as we show in Fig. 1B. This problem becomes exacerbated in the k-way approach, especially as the dimensionality of the eigenspace increases and if the correct number of clusters is not known. An example of this problem is shown in Fig. 1C where the fragmentation of the peppers is caused by this phenomenon. The number of clusters used for the segmentation in Fig. 1C was estimated by minimizing the expression in Eq. (4). The value of this expression as a function of the number of clusters is shown in Fig. 1D. The minimum was obtained for $k = 48$. The results shown in Figs. 1C and D were obtained by using the similarity of pixels in brightness for the $w(i,j)$ matrix in the code made available by the Berkeley group. The value of k was varied from 5 to 50. The reader may wonder how good the result showed in Fig. 1C is as compared to those obtained with other values of k. Actually, this result is among the best one can obtain when using the k-means clustering approach. Although the result is oversegmented, most of the important segments are also present in the image.

As demonstrated by the results shown above, the k-means approach may not always rectify the fragmentation tendency of the Ncut algorithm. This is despite the fact that one can try to find the correct value of k by the minimization of an objective function, such as the one in Eq. (4). This has led us to investigate other approaches to non-parametric clustering in the eigenspace of the affinity matrix. Obviously, each different approach to clustering entails its own method for finding the number of clusters. For example, the method of Koontz and Fukunaga [13] has the advantage of automatically determining the optimal value of k as the data are grouped into clusters. Our work with the Koontz and Fukunaga algorithm shows that it is less sensitive to the Ncut problem introduced above. The comparative results for the "pepper" image are shown in Figs. 2A and B. Note how the new result shown in (A) is less oversegmented than the one shown in (B). Details of how we implemented the Koontz and Fukunaga algorithm are in Section 2.

Although the use of the non-parametric method of Koontz and Fukunaga can help to alleviate the Ncut problem mentioned above, it does not solve this completely. As the reader might have already noted in the result shown in Fig. 2A, there still exist some divisions that are caused by this problem; some of them have been highlighted in Fig. 2C. To solve these remaining cases, we propose to include a

grouping stage that attempts to correct the oversegmentation obtained by the Ncut algorithm. By oversegmentation, we refer to the undesirable, extra segments obtained when using the Ncut criterion.

This grouping stage will thus regroup those areas that would otherwise be divided into two or more partitions. To achieve this though, we will need to modify the weights of our graph with a measure that is tailored to the Ncut problem discussed here. This we will describe in Section 3.

In Section 4, we generalize and simplify the new approach defined in this communication. Experimental results are in Section 5. We will conclude in Section 6.

2. Segmentation as a clustering problem

So far, we have seen how to define the segmentation problem as a graph partitioning one. The problem remains though as how to assign similarity values to the edges in the graph G. Several solutions have been proposed to tackle this. One that was recently shown to work well for a large number of images of different types is defined by

$$w(i,j) = e^{-\frac{\|\mathbf{I}_i - \mathbf{I}_j\|}{\sigma_1^2}} * \begin{cases} e^{-\frac{d(i,j)}{\sigma_2^2}}, & d(i,j) < R, \\ 0, & \text{otherwise}, \end{cases} \tag{5}$$

where $\mathbf{I}_i$ is the brightness value of image $\mathbf{I}$ at pixel i, $d(i,j)$ the Euclidean distance from pixel i to pixel j in the image plane, and σ a control parameter [16,19,21,26].

Once the weights are set, we can use the cut or the Ncut criterion to obtain a partition of the original graph defined by G. For the cut algorithm, this is achieved by first carrying out an eigen-analysis of the Laplacian matrix:

$$Q\mu = \lambda\mu, \tag{6}$$

where $Q = D - W$, with $D(i,j) = \deg(v_i) = \sum_{j=1}^{n} w(i,j)$. W is the matrix whose elements are $w(i,j)$ [1,26]. For the Ncut approach, the eigenvectors are solutions of the generalized eigenvalue decomposition [21]:

$$(D - W)\mu = \lambda D\mu. \tag{7}$$

In both cases, one then retains a certain number, e, of those eigenvectors whose eigenvalues are the smallest except the one whose eigenvalue is zero. Image segmentation is achieved by clustering in such an eigenspace.

A natural way to achieve clustering is by finding the valleys of the densities in the eigenspace. Since, we often cannot assume a parametric form for these densities, a non-parametric method is called for. The method we will use is based on the valley-seeking algorithm advanced by Koontz and Fukunaga [13].

What we accomplish by clustering is the mapping of the set of pixels $\{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$ in the eigenspace to a set of labels $\{z_1, \ldots, z_n\}$; where z_i is an integer between 1 and m (the number of classes) and $m \leqslant n$.

Koontz and Fukunaga non-parametric method is based on the estimate of the density gradient, which is defined as the direction of a sample towards the center

of its cluster. To compute this density gradient, it is common to define a local region area $\Gamma(X)$,

$$\Gamma(X) = \{Y : d(Y,X) \leqslant r\}, \tag{8}$$

where r is the radius of the local region where search for the gradient of the density takes place, and, $d^2(Y,X) = (Y-X)^T H^{-1}(Y-X)$ is a distance measure with metric H. In most cases $H = I$, to represent an Euclidean search.

The local region as defined above has an associated expected vector (*local mean*) which we can define as,

$$M(X) = E[(Y-X)|\Gamma(X)] = \int_{\Gamma(X)} (Y-X)\frac{f(Y)}{u_0}\,\mathrm{d}Y, \tag{9}$$

where

$$u_0 = \int_{\Gamma(X)} f(Y)\,\mathrm{d}Y \approx f(X)v. \tag{10}$$

The term u_0 is used as a normalizing factor given $\Gamma(X)$, v is the volume of $\Gamma(X)$ and $f(\cdot)$ stands for the density function [8]. The local mean can now be used to define the direction of the gradient in Γ. Finally, this can be used to search for the valleys, which are opposite to the gradient.

An easy way to achieve this is to start with an initial classification of the sample vectors and then move the initial "valleys" of each class (or, similarly, the class assignment of each sample vector) opposite to the direction of the gradient of the densities [13].

Formally, given the set of vectors $X = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$ ($\mathbf{x}_i \in [a,b]$), we calculate an initial classification $z_0 = \{z_1, \ldots, z_n\}$ as given by a set of equally separated classes.

For each vector, $\mathbf{x}_i$, we set a local area $\Gamma(\mathbf{x}_i)$ and compute the direction of the gradient of the density. A simple way to accomplish this is by counting the number of samples within the local region $\Gamma(\mathbf{x}_i)$ for each possible class. The vector $\mathbf{x}_i$ is then reclassified as belonging to that class which has the largest number of votes in $\Gamma(\mathbf{x}_i)$. This procedure is repeated until convergence is reached, meaning that no vector is re-assigned to a different class. During this process several classes will merge into one, and at the end the method will have the actual number of classes (clusters) of our representation.

The Euclidean distance between vectors is generally used as a measure of similarity in the graph partitioning approach. Nonetheless, alternatives exist, as for example, the angle between each pair of vectors [4].

Figs. 2A and B show a comparison between the results obtained by using the approach presented above and the k-means clustering algorithm as defined in [14,23]. These results were computed on an eigenspace of 100 dimensions, $r = 0.5$ for the radius of the Γ region, $R = 5$, $\sigma_1 = 0.1$, and $\sigma_2 = 4$.

As to what number of eigenvectors to use, in Fig. 2D we show the segmentation results obtained by clustering (using the method introduced above) in a 10, 20, 30, 40, 50, and 100 dimensional eigenspace ($r = 0.5$ in all cases). We see that the larger the dimensionality, the finer the results. We have experimentally observed, though,

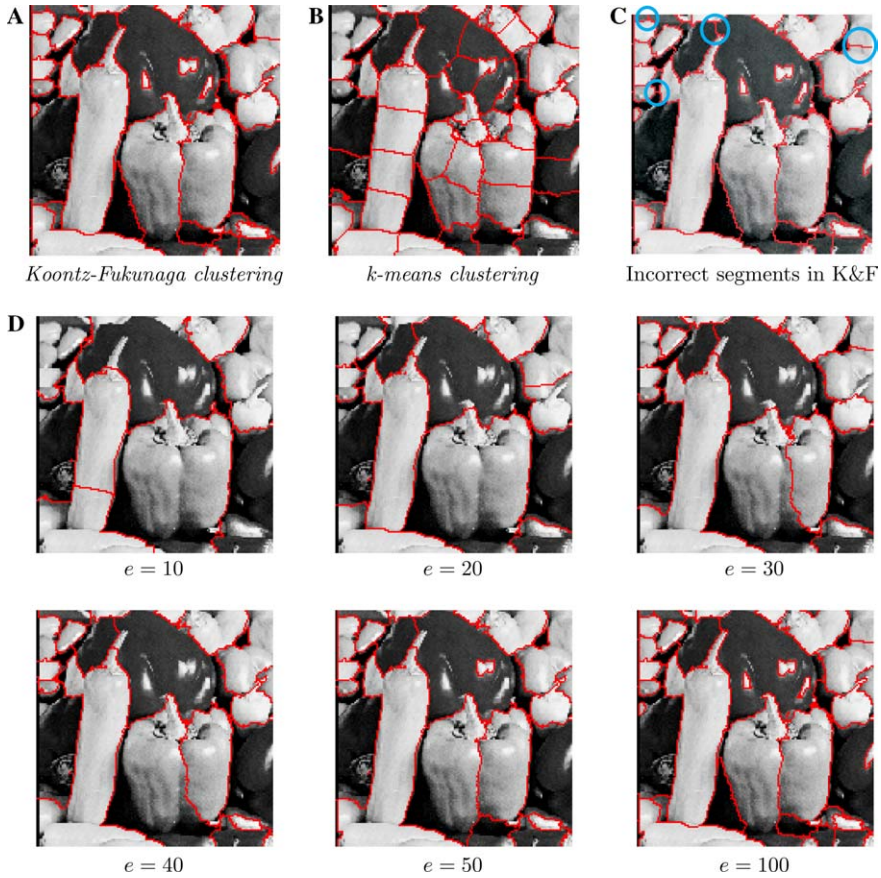

Fig. 2. (A,B) Comparison of the results obtained using the Koontz–Fukunaga clustering algorithm in our approach and the k-means algorithm as described in [14,22]. (C) Some of the problems associated with the Koontz–Fukunaga based implementation are highlighted in the image. (D) Shown here are the segmentation results obtained with Koontz–Fukunaga clustering in the eigenspace spanned by the e smallest eigenvectors. The value e for each segmentation is as shown below the image.

Fig. 3. The results get finer as we modify the value of r.

that increasing the number of eigenvectors over 100 does not result in a much more detailed segmentation for many of our images.

It is interesting to observe (because it is somewhat non-intuitive) from the figures that large blob-like objects do not necessarily show up as individual segments when the eigenspace has low-dimensionality. Such objects, in some cases, are hidden in the higher-dimensions of the eigenspace. So, as with practically all eigenspace based systems, one would want to use the highest possible dimensionality for the eigenspace, subject of course to the limitations imposed by computational burden associated with clustering in very high dimensional spaces.

While the dimensionality of the eigenspace has a great bearing on the granularity of the segments produced, another important influencing factor is the value of r, the radius of the search region. The larger the value of r, the fewer the number of clusters we will have in our e-dimensional space. By the same token, the smaller r is, the finer the results will be. This effect is depicted in Fig. 3 with values of r varying from 1 to 0.5 for a fixed dimensionality of the eigenspace, which was set at 100.

Based on what the theory says and our experimental observations, we believe that one should choose the largest possible value for the dimensionality of the eigenspace and a small enough value for r so as to result in an over-segmentation of the image. Obviously, an over-segmentation is to be preferred to an under-segmentation, because the former contains all of the fragments that when grouped together would yield semantically meaningful objects. This then sets the stage for the next step, which is grouping.

3. Grouping stage

As mentioned in Section 1, the segmentation achieved by Ncut has a tendency to fragment an area of similar brightness into two or more segments. This problem is illustrated by the marked circles in the Ncut-based results in Fig. 2C.

To correct for this over-fragmentation, we will use a grouping stage to the overall segmentation algorithm. As we show in this section, this grouping can be achieved by a second application of the Ncut algorithm, but with each node in the graph representing one segment produced by the first application of Ncut. We now associate with each node the mean gray level of all the pixels that are in the segment corresponding to that node. Using the mean gray level at each node takes advantage of the Ncut's main tendency, which is to group together nodes that are similar.

For the regrouping application of Ncut, we will use the following similarity function for the adjacency nodes:

$$w(i,j) = \begin{cases} e^{\frac{-(\|\mathbf{S}_i - \mathbf{S}_j\|)}{\sigma}}, & \text{if nodes } i \text{ and } j \text{ share boundary pixels in the image,} \\ 0, & \text{otherwise,} \end{cases} \tag{11}$$

where $\mathbf{S}_i$ is the mean value of the brightness of all pixels in segment i.

As was shown previously with the help of the results in Fig. 2C, Ncut even with the non-parametric clustering incorporated in it tends to divide large constant areas into multiple segments. The reader might say that we could improve upon the Ncut results by choosing a larger value for R. But, unfortunately, as shown by the segmentation in Fig. 4, the larger the value of this parameter, the smaller the precision in the delineation of the segments. This is where the second stage of grouping comes in.

As was the case with the first application of Ncut, we are again faced with the question of how many eigenvectors to use in the eigen-representation of the new graph for the second-stage grouping. There is obviously no categorical answer to this question—which was also the case for the first application of Ncut. Nonetheless, it is interesting to explore the output of the second stage grouping as the number of eigenvectors is increased. Obviously again, the larger the number of eigenvectors, the more likely that the final output will correspond to visually different objects in the image.

We will now show the effect of the number of eigenvectors chosen on the quality of the grouping stage. But first note that the number of eigenvectors is now limited to the number of segments obtained after the first application of Ncut. If the first application of Ncut results in m segments, our new affinity matrix (given by Eq. (11)), of size $m \times m$, will possess a maximum of $m - 1$ eigenvectors with associated non-zero eigenvalues. Recall from our previous discussion that it is a property of the affinity matrix that the smallest eigenvalue is always zero.

For this study, we chose for our first-stage processing the segments obtained with the 100 eigenvectors to give the grouping stage a sufficiently fine decomposition of the image. The 100-eigenvector segmentation produced by the first stage was shown earlier in Fig. 2A. The number of segments shown in the figure is 48, implying that the eigenvector decomposition of the grouping-stage adjacency matrix will be limited to a maximum of 47 non-zero eigenvalues. Fig. 5 shows the results obtained from the grouping stage for different values of the number of eigenvectors retained as this number is increased to 30. In this figure, the number of eigenvectors used for the grouping stage is indicated by the symbol e_2. To distinguish this number from the number of eigenvectors used for the first application of Ncut, the previous (first) number will be represented by the symbol e_1.

4. Sub-images for computational efficiency

The main computational burden of the processing described so far is in the first stage—the first application of Ncut. Even a small image, say of size 100×100, results in a $10{,}000 \times 10{,}000$ adjacency matrix—a very large matrix indeed for eigen-analysis. To get around the difficulty of dealing with such large matrices, Malik et al. [14] have suggested first dividing an image into sub-images, applying the Ncut to each sub-image separately, and then applying Ncut to the segments obtained in a second-stage grouping process.

In this section, we will now pull the approach of Malik et al. [14] into our framework and arrive at a scheme that has the advantage of being computationally

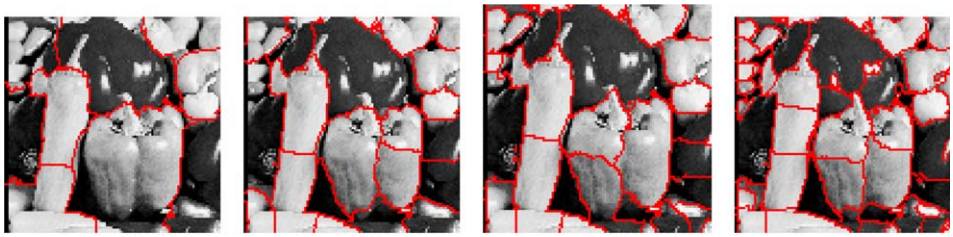

Fig. 4. In an attempt to overcome the problems associated with Ncut, we make larger the value of the neighborhood factor, R, to suppress the division of an area of constant brightness. However, this results in another problem, which is the loss of accuracy in the delineation of the segments.

Fig. 5. Results obtained after the second step with an eigenrepresentation of 100 vectors in the first step and e_2 eigenvectors in the second.

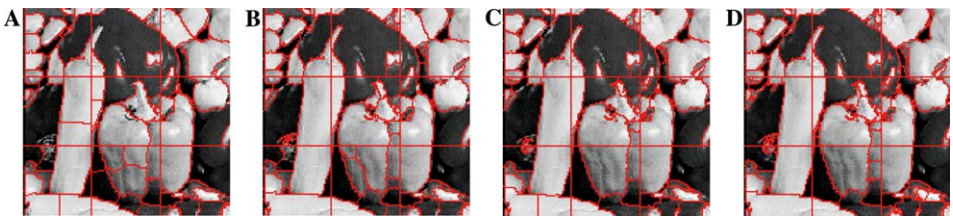

Fig. 6. (A) $e_1 = 10$ and $r = 0.3$, (B) $e_1 = 20$, and $r = 0.4$, (C) $e_1 = 30$ and $r = 0.5$, (D) $e_1 = 40$ and r = 0.5.

Fig. 7. $e_1 = 40$, $r_1 = 0.5$, $e_2 = 30$, and $r_2 = 0.9$.

efficient and at the same time that benefits from our non-parametric clustering and grouping criterion.

Fig. 6 illustrates this approach. Note that the boundaries that connect each of the sub-images are now preserved in the output of the first application of Ncut. Our second-stage grouping treats the sub-image boundaries like any other boundaries between the segments produced by the first stage. The rest of the processing proceeds just as before. The final result is shown in Fig. 7.

5. Experimental results

To explain the various steps of our approach, the discussion so far used only one image. In this section, we show several additional results on different types of images. We will compare the segmentations obtained using our approach with those obtained with the method described in [14,23]. As we did before, all the results shown will use an affinity matrix that measures pixel similarities on the basis of the brightness levels.

All of the experimental work shown in this section first partitions an image into 2×3 sub-images. In light of our previously mentioned rationale for partitioning an image into sub-images before the first application of Ncut, we retain only the smallest 50 eigenvector for this phase of our overall approach. Our experiments show that retaining fewer eigenvectors at this stage noticeably degrades the segmentations, in the sense that even large segments may disappear. And retaining more eigenvectors does not significantly improve the quality of the final segmentation.

For the Koontz and Fukunaga clustering algorithm for the first application of Ncut, we used $r_1 = 0.3$. This value for the radius works well for the dimensionality of 50 for the eigenspace. Ordinarily, the larger the dimensionality of the space used for data representation, the larger the mean distance between the data points. This means that the value of r would increase with the dimensionality of the eigenspace.

This brings us to the experimental parameters used for the second application of Ncut for the grouping stage. As the reader will recall from Section 3, the full dimensionality of the eigenspace here is limited by the total number of segments produced by the first application of Ncut in all of the 6 sub-images. In the comparative results shown, we have used the dimensionality of 100 for all the images. However, we have also shown comparative results when all of the eigenvectors with non-zero eigenvalues are retained.

Fig. 8A shows three images on which we will compare the performance of Koontz–Fukunaga clustering with k-means clustering. Fig. 8B shows the results produced for the images using the Koontz–Fukunaga algorithm with the maximum number of eigenvectors retained in the grouping stage. Fig. 8C shows the results obtained when only 100 eigenvectors are retained. And, to compare, Fig. 8D shows the results produced with the k-means approach as described in [14,23].

Finally, to show the versatility achieved when clustering is carried out with the Koontz–Fukunaga algorithm, we show additional comparative results in Figs. 9 and 10. For the comparisons shown in these figures, we have used only 100 eigenvectors in the grouping-stage application of Ncut for our approach.

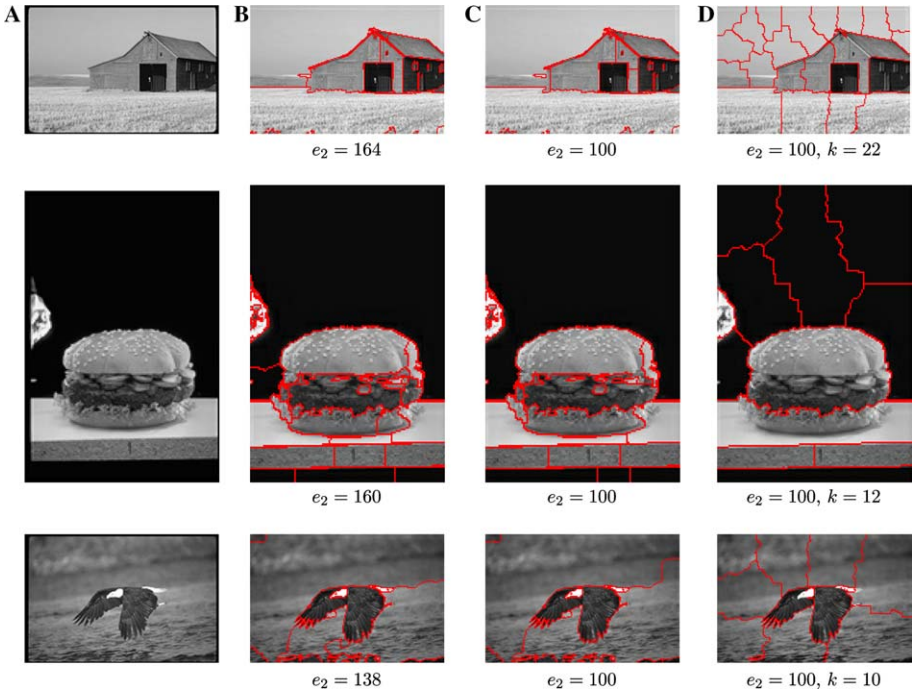

Fig. 8. (A) Original images. (B) Segmentation with Koontz–Fukunaga clustering using all eigenvectors corresponding to non-zero eigenvalues and with $r_2 = 2.0$. (C) Same as in (B) but with the number of eigenvectors e_2 equal to 100. (D) Segmentation results obtained with k-means clustering and with $e_2 = 100$.

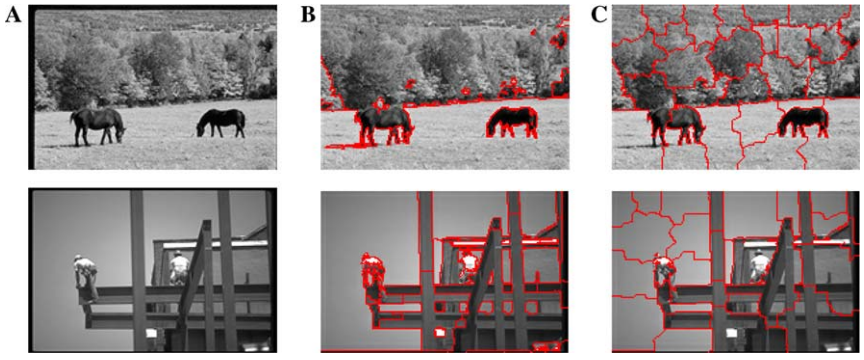

Fig. 9. For the two images shown, it is necessary to extract several segments corresponding to highly localized detail. (A) Original images. (B) Segmentations obtained using Koontz–Fukunaga clustering with $e_2 = 100$. (C) Segmentations obtained using k-means clustering with $e_2 = 100$.

6. Conclusions

This paper presented an alternative implementation of the k-way Ncut graph-partitioning approach to image segmentation [14,21,23]. In our implementation,

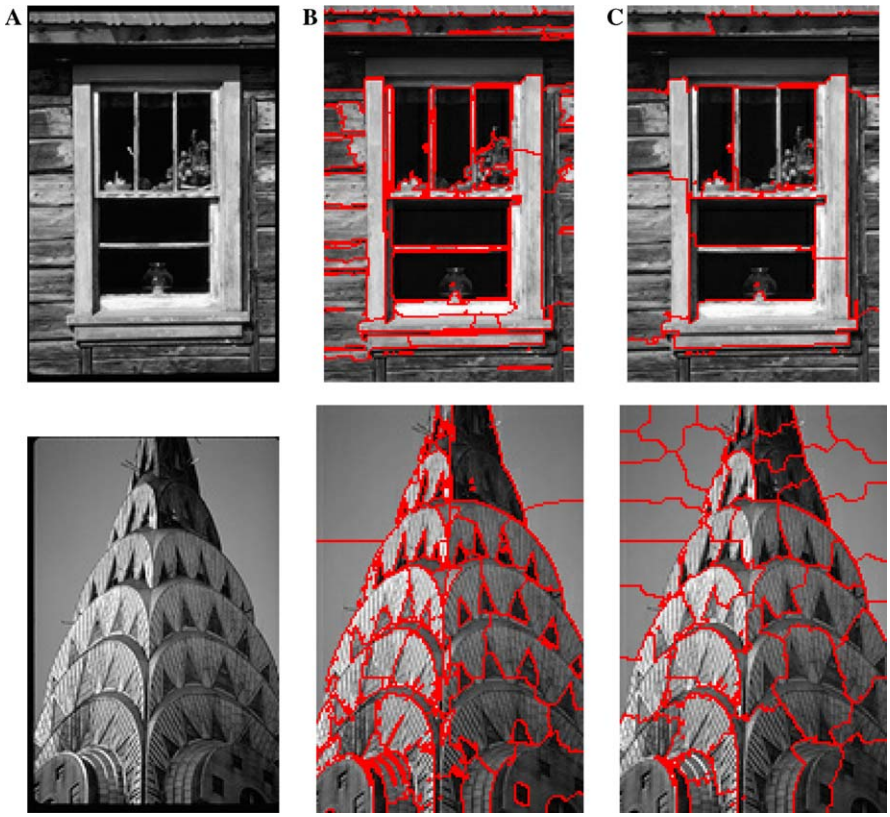

Fig. 10. These two images possess a large number of visually similar sub-structures. (A) Original images. (B) Segmentations obtained using Koontz–Fukunaga clustering with $e_2 = 100$. (C) Segmentations with k-means clustering with $e_2 = 100$.

the clustering in the eigenspace is carried out by using the Koontz–Fukunaga non-parametric algorithm, as opposed to the k-means algorithm used in [14,23]. Additionally, we use a two-stage application of Ncut, with the second stage meant for the grouping of the fragments produced by the first application of Ncut. The affinity matrix for the grouping stage uses the similarity of mean values for the image fragments.

Acknowledgments

We thank the Berkeley group for making their code available to us. This research was partially supported by NSF (National Science Foundation) and NIH (National Institute of Health).

References

[1] C.J. Alpert, A.B. Kahng, Recent developments in netlist partitioning: a survey, Integration: VLSI J. 19 (1995) 1–81.

[2] C.J. Alpert, A.B. Kahng, S. Yao, Spectral partitioning with multiple eigenvectors, Discrete Appl. Math. 90 (1999) 3–26.

[3] E.R. Barnes, An algorithm for partitioning the nodes of a graph, SIAM J. Alg. Dic. Math. 3 (1982) 541–549.

[4] P.K. Chan, M.D.F. Schlag, J. Zien, Spectral k-way ratio cut partitioning and clustering, IEEE Trans. CAD (1994) 1088–1096.

[5] C.H.Q. Ding, X. He, H. Zha, M. Gu, H.D. Simon, A min-max cut algorithm for graph partitioning and data clustering, Proc. IEEE Int. Conf. Data Mining (2001).

[6] J. Frankle, R.M. Karp, Circuit placements and cost bounds by eigenvector decomposition, Proc. IEEE Int. Conf. Comput. Aided Des. (1986) 414–417.

[7] M. Fiedler, Algebraic connectivity of graph, Czech. Math. J. 23 (1973) 298–305.

[8] K. Fukunaga, Introduction to Statistical Pattern Recognition, second ed., Academic Press, New York, 1990.

[9] Y. Gdalyahu, D. Weinshall, M. Werman, Self-organization in vision: stochastic clustering for image segmentation, perceptual grouping, and image database organization, IEEE Trans. Pattern Anal. Mach. Intell. 23 (10) (2001) 1053–1074.

[10] S.W. Hadley, B.L. Mark, A. Vannelli, An efficient eigenvector approach for finding netlist partitions, IEEE Trans. CAD 11 (1992) 885–892.

[11] L. Hagen, A.B. Kahng, New spectral methods for ratio cut partitioning and clustering, IEEE Trans. Comput. Aided Des. 11 (1992) 1074–1085.

[12] K.M. Hall, An r-dimensional quadratic placement algorithm, Manag. Sci. 17 (1970) 219–229.

[13] W.L.G. Koontz, K. Fukunaga, A nonparametric valley-seeking technique for cluster analysis, IEEE Trans. Comput. 21 (1972) 171–178.

[14] J. Malik, S. Belongie, T. Leung, J. Shi, Contour and texture analysis for image segmentation, Int. J. Comp. Vis. 43 (1) (2001) 7–27.

[15] B. Mohar, in: Y. Alavi, G. Chartrand, O.R. Oellermann, A.J. Schwenk (Eds.), The Laplacian Spectrum of Graphs, In Graph Theory, Combinatorics and Applications, vol. 2, Wiley, New York, 1991, pp. 871–898.

[16] P. Perona, W.T. Freeman, A factorization approach to grouping, Proc. Eur. Conf. Comput. Vis. (1998) 655–670.

[17] A. Sanfeliu, R. Alquézar, J. Andrade, J. Climent, F. Serratosa, J. Vergés, Graph-based representations and techniques for image processing and image analysis, Patt. Recogn. 35 (2002) 639–650.

[18] S. Sarkar, K.L. Boyer, Computing Perceptual Organization in Computer Vision, World Scientific, Singapore, 1994.

[19] S. Sarkar, K.L. Boyer, Quantitative measures of change based on feature organization: eigenvalues and eigenvectors, Proc. IEEE Conf. Comput. Vis. Patt. Recogn. (1996) 478–483.

[20] S. Sarkar, P. Soundararajan, Supervise learning of large perceptual organization: graph spectral partitioning and learning automata, IEEE Trans. Patt. Anal. Mach. Intell. 22 (5) (2000) 504–525.

[21] J. Shi, J. Malik, Normalized cuts and image segmentation, IEEE Trans. Patt. Anal. Mach. Intell. 22 (8) (2000) 888–905.

[22] P. Soundararajan, S. Sarkar, An in-depth study of graph partitioning measures for perceptual organization, IEEE Trans. Patt. Anal. Mach. Intell. 25 (6) (2003) 642–660.

[23] D. Tal, J. Malik, "Combining color, texture and contour cues for image segmentation, Preprint, 2001.

[24] S. Wang, J.M. Siskind, Image segmentation with minimum min cut, Proc. Int. Conf. Comput. Vis. (2001).

[25] S. Wang, J.M. Siskind, Image segmentation with ratio cut, IEEE Trans. Patt. Anal. Mach. Intell. 25 (6) (2003) 675–690.

[26] Z. Wu, R. Leahy, An optimal graph theoretical approach to data clustering: theory and its application to image segmentation, IEEE Trans. Patt. Anal. Mach. Intell. 15 (1993) 1101–1113.