Community Detection for Hierarchical Image Segmentation

Arnaud Browet, P.-A. Absil, and Paul Van Dooren

ICTEAM Institute, Université catholique de Louvain, Louvain-la-Neuve, Belgium
{arnaud.browet,paul.vandooren}@uclouvain.be,
absil@inma.ucl.ac.be

Abstract. In this paper, we present a new graph-based technique to detect segments or contours of objects in a given picture. Our algorithm is designed as an approximation of the Louvain method that unfolds the community structures in a large graph. Without any a priori knowledge on the input picture, relevant regions are extracted while the optimal definition of a contour, depending on the user or the application, can be tuned using parameters. The communities found are also hierarchical allowing to find subregions inside an object. We present experimental results of our method on real images.

Keywords: Image segmentation, community detection, modularity optimization.

1 Introduction

How many objects are present in a picture? This simple question from a perception point of view (depending on what one considers to be an object) has raised a lot of research for the last decades and remains a very challenging problem for a computer. There are many applications where this consideration takes place: discovering abnormal shadows on a CT or PET scan for tumor detection, detection of people and objects from images of surveillance cameras or from a camera at the front of a car in the context of collision detection, etc.

The answer is never unique in image segmentation because defining the objects of interest is user or application dependent. However, we want to address the question of image segmentation without any a priori knowledge about the shape, the position or the number of objects displayed in the input picture. Nevertheless, we will keep a number of tuning parameters to optimize the method depending on some properties of the picture, like the number of pixels or their range. Based on a graph built from the image, we will show that finding the relevant structure of objects is possible using the optimization of a scalar function called the modularity. This measure was introduced by Newman and Girvan [16] and represents the quality of clusters defined on the nodes of the graph. Unfortunately, like the definition of clusters, this measure is combinatorial and the optimization of this criterion cannot be done in polynomial time. We will present a greedy algorithm that provides a good approximation of the optimal modularity. This algorithm will allow us to define communities of nodes in the graph which lead to coherent groups or subgroups of pixels in the image.

The paper is organized as follows: in section 2 we review some classical graph-based techniques for image segmentation and introduce our notations. Then in section 3 we review the concept of modularity for community detection in large graphs and describe our algorithm to approximate the optimal modularity. In section 4 we show some experimental results on real images. Finally, section 5 concludes with the main results of this paper and proposes future work.

2 Related Work

Since the number of techniques in computer vision is quite large, each one coming from various fields (histograms metric [21], watershed techniques [3], active contours [13,14], manifold learning [12,19], etc), we will restrict our review to graph-based techniques. Those procedures generally build a graph G where each node represents a pixel in the input picture. Since the number of pixels is very large, even for low resolution pictures, ideas have been proposed to reduce the number of nodes in the graph to a representative subsample of pixels or regions, for example in [2] or see other references therein.

Each weighted edge of the graph G represents the similarity between a pair of pixels and is stored in the weighted adjacency matrix W. Without a priori knowledge about the components of the input image, a classical way to define an edge weight is

$$w_{ij} = \begin{cases} e^{\frac{d(i,j)^2}{\sigma_x^2}} e^{\frac{|F(i)-F(j)|^2}{\sigma_i^2}} & \text{if} \quad d(i,j) < d_{max}, \\ 0 & \text{otherwise}, \end{cases} \tag{1}$$

where $d(i,j)$ is the distance between pixels i and j (e.g. the Euclidean or the Chebyshev distance) and $F(i)$ is a feature vector evaluated at pixel i. This feature vector can be for example the scalar intensity value for gray scaled images or the HSV transform for color images. The edge weight is controlled by the user defined parameters σ_x, σ_i and d_{max}.

Based on this undirected weighted graph, classical methods [5,18,20,22] try to find a selection of edges, called a cut, to optimize a criterion. A cut is defined as a set of edges that creates disconnected components when removed from the graph. The cost function is in general the total weight of the removed edges with an additional scaling that penalizes the creation of very small components: in the ratio cut the scale is the dimension of the components (Cox *et al.*[5]), in the minimum mean cut the scale is the number of cut edges (Wang & Siskind [20]) and in the normalized cut the scale is the internal similarity of the components (Shi & Malik [18]). Another efficient method has been proposed by Felzenszwalb and Huttenlocher [7] that extracts regions in the graph minimizing what they call internal difference of regions while maximizing the external difference between regions. Unfortunately, in general these methods need to know the number of objects of interest in the input picture or they have to set an arbitrary threshold on the minimum value of the cut criterion and this may lead to an inappropriate segmentation. In particular, optimizing the normalized cut criterion is known to be NP-hard [18] but a continuous relaxation can be solved in polynomial time using spectral graph theory. Some recent works [1,9] propose to reduce the computational cost of such methods by computing an approximation of the optimal cut criterion using the incomplete Cholesky decomposition.

Those methods try to split the large pixel graph into salient regions by defining boundaries between the regions. On the other hand, one can consider the opposite way of grouping adjacent connected nodes, defining the salient regions and letting the boundaries appear by themselves. We propose to analyze this idea by considering the previously constructed graph as a large (social) network divided in communities. Communities as introduced by Newman and Girvan [10] are defined as sets of highly connected nodes with only a few or light connections between distinct groups. This informal definition is obviously not sufficient to identify the community structure in a graph. Therefore Newman and Girvan [16] introduce a scalar function Q called the modularity to evaluate the quality of a community structure in a graph:

$$Q = \frac{1}{2m} \sum_{i,j=1}^{n} (w_{ij} - N_{ij})\, \delta(C_i, C_j) \ , \tag{2}$$

where n is the number of nodes in the graph, w_{ij} is the edge weight as previously defined in (1), N_{ij} is a null model between nodes i and j, C_i is the community of node i and $\delta(C_i, C_j) = 1$ if nodes i and j are in the same community and 0 otherwise. Here m is the sum of the weights of all the edges in the graph and is a scaling parameter ensuring that $Q \in [-1, 1]$. The null model N_{ij} is designed to assess the strength of an edge and can be defined as the expected weight of the edge between nodes i and j, given that the degree of each node is known:

$$N_{ij} = \frac{k_i k_j}{2m} \ . \tag{3}$$

Here k_i is the weighted degree of node i i.e. $k_i = \sum_{j=1}^{n} w_{ij}$. The null model presented here is just a straightforward extension from an unweighted framework. If one considers the selection of stubs (half-edges), the probability to pick up a stub from node i is $p_i = k_i/2m$ since there are k_i outgoing stubs from node i out of $2m$ stubs in total. The probability to have an edge between i and j is then defined as the probability to select a stub from node i and a stub from node j, so it is proportional to $p_i p_j$ as presented. For a more elaborate development of the null model, see [8].

It follows that if the actual edge weight w_{ij} is larger than the expected edge weight N_{ij}, nodes i and j are strongly connected and assigning them to the same community will lead to a positive modularity gain.

One can see that maximizing the modularity over all possible community partitions (including the number of communities) is a combinatorial problem and cannot be solved in polynomial time, hence the interest for fast algorithms that build an approximation of the optimal modularity.

In the next section, we first review a greedy algorithm, called the Louvain method, to approximate the optimal modularity of a graph and then we will explain the refinements that we brought to this method to segment an input picture.

3 Optimizing the Modularity

There exist different techniques to uncover the community structure in a graph using modularity maximization (a very detailed review has been made by Fortunato [8]). We choose to use a modification of the Louvain method because this method is very successful in terms of computational cost and quality of the detected communities.

3.1 The Louvain Method

The Louvain method introduced by Blondel *et al.* [4] is a greedy algorithm to uncover community structures in a weighted graph using a local decision for each node. This algorithm produces hierarchical communities by recursively aggregating smaller communities.

At the initialization, each node of the graph, i.e. each pixel, defines its own community. At this step, $Q < 0$ since we choose $w_{ii} = 0$ by definition so the modularity is smaller than in the situation where all the nodes are in the same community ($Q = 0$). Then, a node is randomly chosen and each gain of modularity of grouping this node to any of its neighboring communities is computed. The node is finally assigned to the community with the maximal (positive) gain. The procedure iterates over randomly chosen nodes until no positive gain can be found (each node can be selected more than once since a node can be taken out of its pre-assigned community to check if no other community assignment can produce a larger modularity gain). After convergence, all the communities found are aggregated to form a new graph: each community becomes a single node, with a self-loop computed as the sum of the weights of all the internal edges, while the edges between the new nodes are computed as the sum of the old external edges. This two-steps procedure is then recursively applied to each of the aggregated graphs produced until no positive gain can be found for any isolated node (figure 1 illustrates the algorithm on a synthetic graph). Note that optimizing the modularity on each aggregated graph is equivalent to an optimization of the modularity based on the initial graph i.e. the modularity of a community structure on the aggregated graph has exactly the same value as the modularity of the induced underlying communities on the initial graph. This property is very valuable since we want to optimize the community structure of the initial picture-based graph without losing information during the aggregation step.

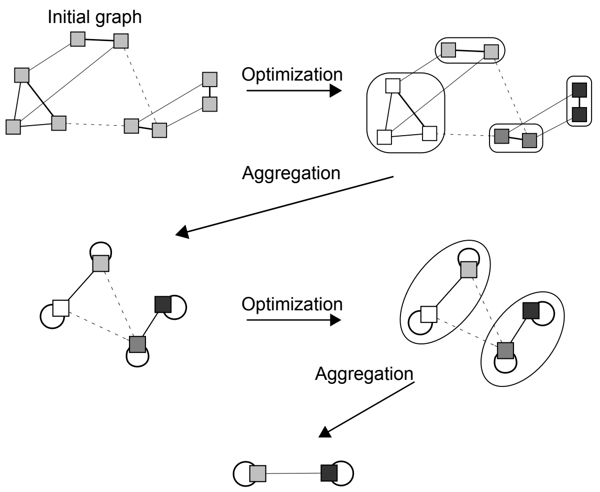


Fig. 1. Representation of the steps applied by the Louvain method on a synthetic graph. At each loop, the algorithm first computes an optimization of the modularity on a graph; then it aggregates the graph according to the communities found.

3.2 A Modified Louvain Method

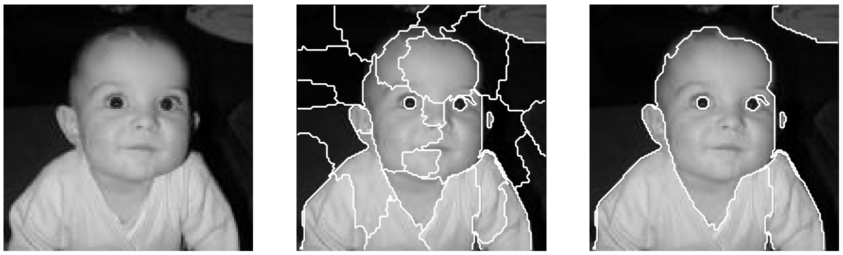
The Louvain method works well in practical situations but is time consuming at the first iteration (considering the problem of real time video tracking for example) because of the loop over all node. We propose a modification of this algorithm to allow faster computation.

We describe here what we call a basic iteration of the algorithm. First, we compute a gain matrix G that contains the gain of assigning each node with one of its neighbor. This gain matrix is easily computed by

$$G = \frac{1}{2m}\left(2\,W - \frac{1}{m}kk^T\right) \ , \tag{4}$$

where k is the vector of degrees. This formulation is very simple because at the beginning of each iteration the communities contain only a single node. Note that the computation of kk^T can be restricted to the indices of non zero elements of W since we are only interested in positive gains.

For each node, we derive an assignment with its best neighbor, defined by the maximum value on each row of the gain matrix G. Then, we consider a directed graph defined by these assignments and we construct the communities as the connected components of this graph. Since the best neighbor assignment can obviously be non-symmetric, the connected components may become very large. This leads to the creation of large communities but it can have the side effect that the negative contribution of the null model becomes larger than the positive contribution of the edge weight, leading to a negative modularity gain. We will



(a) Input picture (b) Segmentation using classical modularity (c) Segmentation using weighted modularity

Fig. 2. Segmentation of a baby face (130 × 132 pixels) using classical and weighted modularity: $\sigma_x = 1.2$, $\sigma_i = 10$, $d_{max} = 3$ and $d_\Lambda = 15$. There are 34 regions segmented by the classical modularity and 8 regions segmented by the weighted modularity.

illustrate this phenomenon afterward. In this situation, we recursively remove, from its community, the node with the smallest (negative) gain until the total gain becomes positive.

Finally, once the communities have been defined, they are aggregated to create a new smaller graph.

We iterate this basic iteration until the gain matrix does not contain any positive entry anymore. This algorithm is deterministic and allows, at each step, to assign all the nodes at the same time to a community, leading to a clear improvement in term of computing time.

Fig.2(a) and 2(b) show the results of the segmentation on a real image using our community detection algorithm. The results of Fig.2(c) will be discussed later.

One can see that all the regions segmented by the algorithm are coherent but also that the objects of interest are oversegmented (6 regions for the shirt and 7 for the face). This oversegmentation comes from the fact that modularity maximization algorithms cannot yield communities with long chains of nodes. In this context, large regions in the picture (including the background), even with similar intensity levels, cannot be merged together: the distance d_{max} in (1) is small compared to the size of the picture (due to memory limitation), leading to a huge number of disconnected pairs of pixels. Defining a community that contains a large distance between pixels will lead to a smaller (or even negative) modularity because of these disconnected pixels. To illustrate this, suppose that we split the background of an image into 2 adjacent non-overlapping regions r_1 and r_2 and that we denote by ∂r_i the border of width d_{max} of the region i adjacent to region j as depicted in Fig.3. If we denote by Q_r the contribution in the modularity of a community defined by the region r

$$Q_r = \frac{1}{2m} \sum_{i,j \in r} \left(w_{ij} - \frac{k_i k_j}{2m} \right) ,$$

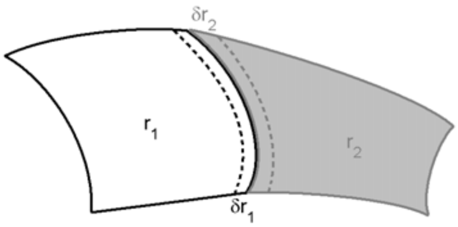


Fig. 3. Representation of 2 adjacent non-overlapping regions with their respective adjacent border

then one can see that

$$Q_{r_1 \cup r_2} = Q_{r_1} + Q_{r_2} + \frac{1}{m} \sum_{\substack{i \in r_1 \\ j \in r_2}} \left(w_{ij} - \frac{k_i k_j}{2m} \right) . \tag{5}$$

The gain of modularity $\varDelta Q_{r_1 \leftrightarrow r_2}$ of grouping r_1 and r_2 together is then given by

$$\begin{aligned} \varDelta Q_{r_1 \leftrightarrow r_2} &= Q_{r_1 \cup r_2} - Q_{r_1} - Qr_2 \\ &= Q_{\partial r_1 \cup \partial r_2} - Q_{\partial r_1} - Q_{\partial r_2} \\ &\quad - \frac{1}{m} \sum_{\substack{i \in r_1 \setminus \partial r_1 \\ j \in r_2}} \frac{k_i k_j}{2m} - \frac{1}{m} \sum_{\substack{i \in \partial r_1 \\ j \in r_2 \setminus \partial r_2}} \frac{k_i k_j}{2m} , \end{aligned} \tag{6}$$

and if the number of pixels in both regions is large, the last two negative terms of this equation dominate, leading to a smaller global modularity.

There are at least two ways to tackle this problem of resolution. First, one can increase the distance d_{max} which in turn increases the size of the border ∂r_i and therefore reduces the number of nodes in $r_i \setminus \partial r_i$. While this is a good theoretical solution that may work on very small images, it is not suitable for typical images; one can check that the number of neighbors for each pixel grows as $O(d_{max}^2)$, making this solution impossible in practice, due to the amount of memory needed to compute W. Another possible solution is to modify the null model (3) in such a way that it takes into account the distance inside a community, defining a weighted version of the modularity. We pursue this idea in the next section.

3.3 The Weighted Modularity

To avoid oversegmentation, the null model should be defined such that it takes into account the fact that an object can be spread over a large area of the picture *i.e.* the computation of the modularity in (2) should be mainly local. In their paper, Reichardt and Bornholdt [17] introduce a constant weighting parameter $\lambda < 1$ in the null model

$$N_{ij} = \lambda \frac{k_i k_j}{2m} \ ,$$

and show that it produces larger communities. Some other scalings have also been proposed by Delvenne, Yaliraki and Barahona [6]. Based on these observations, we proposed to define the null model as

$$N_{ij} = \Lambda_{ij} \frac{k_i k_j}{2m} \ , \tag{7}$$

where the matrix Λ is defined as

$$\Lambda_{ij} = \begin{cases} 1 \text{ if } d(i,j) \leq d_\Lambda \ , \\ 0 \text{ otherwise } \ , \end{cases} \tag{8}$$

where d_Λ is a parameter depending on the size of the picture and the size of the objects.

The computation of the aggregation step of the algorithm is straightforward in the context of the classical modularity but it requires some basic matrix computations for the weighted modularity. Let us suppose that the communities are defined by a matrix

$$C \in \{0,1\}^{p \times n} : C_{ij} = 1 \text{ if community } i \text{ contains node } j \ ,$$

where p is the number of communities and n the number of pixels. The aggregated graph $\widetilde{G}$ is defined by

$$\begin{aligned} \widetilde{W} &= C\, W\, C^T, \\ \widetilde{k} &= \widetilde{W}\, \mathbb{1}_p = C\, W\, \mathbb{1}_n = C\, k, \end{aligned}$$

since we assumed non overlapping regions, thus $C^T\, \mathbb{1}_p = \mathbb{1}_n$, where $\mathbb{1}_p$ is the vector of size p of all ones.

In the framework of weighted modularity, we still want to keep the property that optimizing the modularity on the initial graph is equivalent to optimizing to modularity of any aggregated graph (up to a constraint). This can still be done if the weight matrix Λ in the aggregated graph is defined by

$$\widetilde{\Lambda} = (\widetilde{K})^{-1} \left(C\, K\, \Lambda\, K\, C^T\right) (\widetilde{K})^{-1}, \tag{9}$$

where K is the diagonal matrix defined by $K = diag(k)$.

Table 1. Parameters used for the segmentation presented in Fig.4

row 1	$\sigma_x = \sqrt{2}$, $\sigma_i = 5$, $d_{max} = 3$, $d_\Lambda = 20$
row 2	$\sigma_x = \sqrt{2}$, $\sigma_i = 3$, $d_{max} = 2$, $d_\Lambda = 20$
row 3	$\sigma_x = \sqrt{2}$, $\sigma_i = 2$, $d_{max} = 2$, $d_\Lambda = 25$
row 4	$\sigma_x = \sqrt{2}$, $\sigma_i = 4$, $d_{max} = 3$, $d_\Lambda = 20$
row 5	$\sigma_x = 2$, $\sigma_i = 2$, $d_{max} = 4$, $d_\Lambda = 30$

One can observe that this definition is consistent with the classical modularity: if we define $\Lambda = \mathbb{1}_n \mathbb{1}_n^T$ then $\widetilde{\Lambda} = \mathbb{1}_p \mathbb{1}_p^T$.

The result of the weighted modularity applied to our test picture of the baby face is represented in Fig.2(c) for $d_\Lambda = 15$. There are 8 regions defined by the algorithm and they have very good visual relevance.

4 Experimental Results

We ran our algorithm on a set of pictures taken from the Berkeley image segmentation database [15] and found relevant contours of objects when the parameter of the method were correctly defined. Some results are displayed in Fig.4 along with the initial pictures and human segmentation benchmarks. The different pictures are presented in increasing level of complexity according to the benchmark, and the parameters used for each segmentation are presented in table 1. One can see that our algorithm is able to correctly identify the general contour of the objects in each picture but tends to have some difficulties when defining a boundary in low gradient regions like the neck of the camel in row 3. There is also more merging of communities that might be considered, for example in row 4 in the background behind the violinist or in row 5 for the large rock at the right side. Those mergings are not allowed by the algorithm because the communities do not have enough edges between each other but we plan to investigate this question by adding a penalty for the smoothness of the contours.

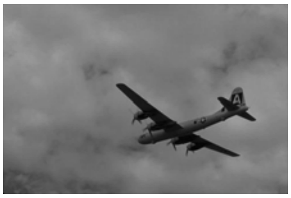
Fig.5 shows the result on a picture of elephants. One can see that the algorithm found 7 main regions, uncovering the 2 elephants, and some minor communities with less than 10 nodes due to image quality and artefacts, as shown in Fig.5(a). Those small communities can be easily postprocessed and merged with the surrounding communities.

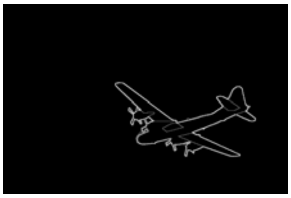
The algorithm requires about 2 seconds to segment the 320×480 pictures on a Intel Core 2 Quad, 2.5 GHz with 4 Go ram, using Matlab.

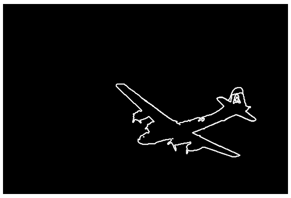
Another important feature of our method is that the detected communities are hierarchical, meaning that each community is only the aggregation of smaller communities. This allows us to extract smaller parts of objects contained in larger areas of the picture. As an example, Fig.5(b) shows the communities found at the penultimate aggregation step. This step discovers finer boundaries between relevant parts of the picture such as a distinction between the two animals.

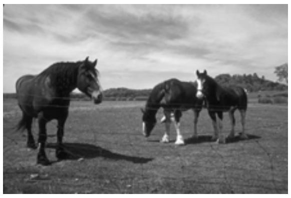
The decrease of the number of communities is exponential in the number of basic iterations as shown in Fig.6, independently of the figure or the number of pixels. Knowing that the assignment on each step is done in $\mathcal{O}(E)$ where E is the number of edges in the graph $\widetilde{G}$, this allows the segmentation of the input image to be done in a reasonable amount of time.

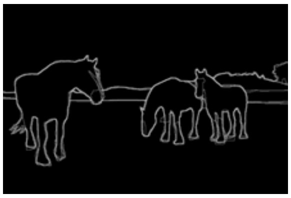
While we observe good segmentation results, we also notice that the communities found can change dramatically with a minor modification of the parameters (σ_x, σ_i, d_{max} or d_Λ). This sensitivity to the parameters is observed

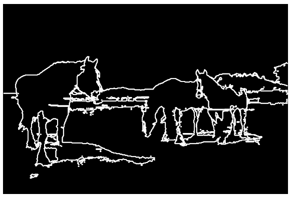

(a) Input picture

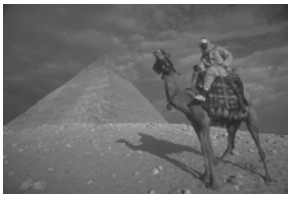

(b) Human benchmark

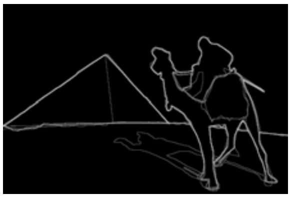

(c) Our segmentation

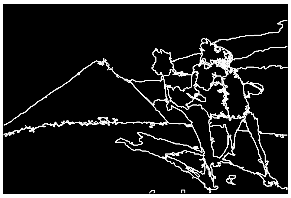

(d) Input picture

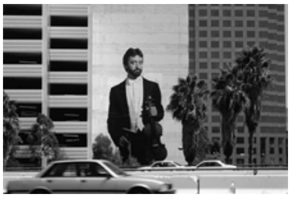

(e) Human benchmark

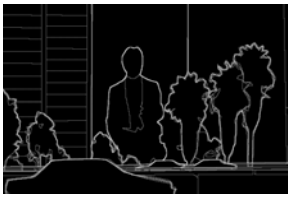

(f) Our segmentation

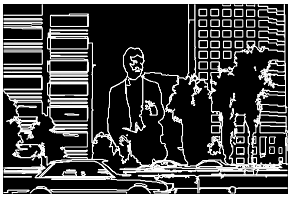

(g) Input picture

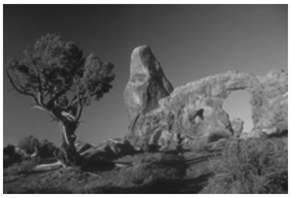

(h) Human benchmark

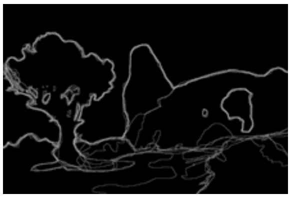

(i) Our segmentation


(j) Input picture


(k) Human benchmark


(l) Our segmentation


(m) Input picture


(n) Human benchmark

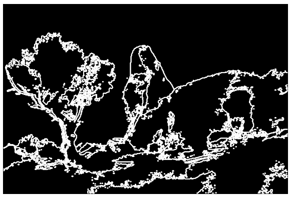
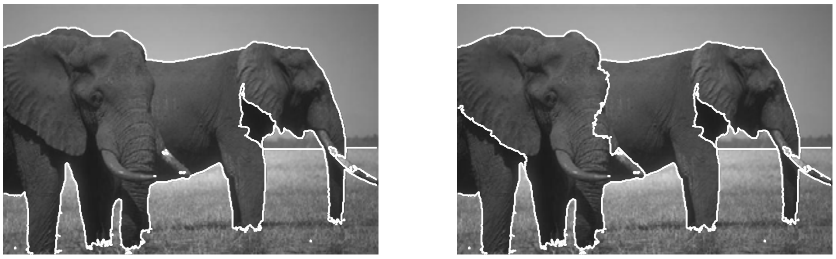

(o) Our segmentation

Fig. 4. Segmentation results on some pictures from the Berkeley image segmentation database. Each row presents the initial picture, the human segmentation and the result found by our algorithm.



(a) Segmentation using weighted modularity (b) Segmentation 1 step before the final aggregation step

Fig. 5. Segmentation of 2 elephants (320 × 480 pixels) using weighted modularity: $\sigma_x = 5$, $\sigma_i = 9$, $d_{max} = 3$ and $d_\Lambda = 25$. In 5(a), there are 7 main regions segmented (and some small communities, less than 10 pixels, due to imaging artefact) representing the background and the animals. In 5(b), the segmentation at the penultimate aggregation step is displayed showing that there are in fact 2 distinct elephants.

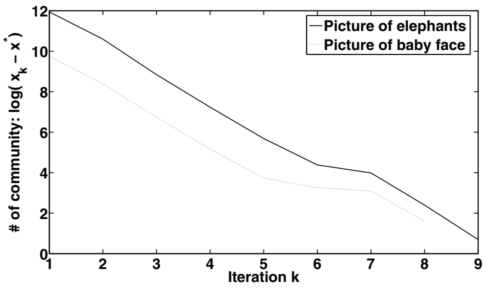


Fig. 6. Evolution of the number of communities showing the exponential decrease in the number of basic iterations

in most segmentation methods but it seems to affect strongly our results. One can then ask if our method is able to derive good parameters based on the modularity value. Unfortunately, it has already been shown by Good *et al.* [11] that the modularity cannot be compared across different graphs. We also observed that the "optimal"[1] modularity value exhibits the shape of a plateau when the parameters vary as represented in Fig.7. It is not possible to define good parameters leading to a well clustered graph based on this criterion. Because of the high sensitivity of the method to parameters selection, one needs to further investigate this question to better understand the dependencies between parameters and derive good selection rules. This is a topic for future paper.

[1] Optimal in the sense of the best community definition found by the method.

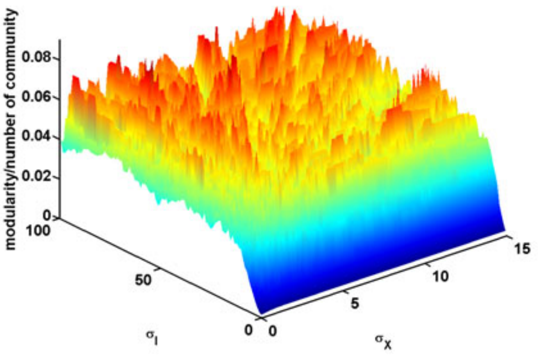


Fig. 7. Modularity, weighted by the number of communities, for different values of the parameters σ_x and σ_i. Since changing the parameters leads to different graphs, the values of modularity cannot be compared.

5 Conclusion

Image segmentation is a difficult problem but we show that when a graph is constructed from a given image using well chosen parameters, extracting communities from this graph can produce salient contour detection of the objects inside the picture. The community definition can be quickly computed by considering an approximation of the Louvain method. This greedy algorithm optimizes a criterion called the modularity that produces relevant community structures.

Classical modularity suffers from a problem of resolution imposing that the radius of communities should stay small. This limitation generally results in oversegmentation of the input picture. We proposed an alternative definition, called weighted modularity, to tackle this problem by weighting the null model according to a smaller radius. Based on this new criterion, we showed that good segmentation can be achieved in a reasonable amount of time. The communities defined are also hierarchical, allowing to uncover smaller entities inside larger objects or shapes.

Parameters selection for the edge weight is a crucial problem and unfortunately, modularity optimization is not able to define the parameters leading to a good pixel clustering, since modularity exhibits the shape of a plateau when it is compared across different weight matrices W for the same image. Another parameter selection can be discussed: as mentioned, a reduction of the oversegmentation can be achieved by using a larger distance radius d_{max} or by considering weighted modularity with a radius d_{Λ}. These parameters do not seem to be independent and some work has to be done in this direction to better understand the relationship between them.

An interesting topic for future research is the possibility to use an approximation of the matrix Λ. At the early step of the method, computing this matrix can be time consuming. This can be greatly improved by considering a low rank approximation of Λ if the segmentation results stay accurate.

Finally, we also plan to extend the method to video tracking allowing to track multiple objects on consecutive video frames.

Acknowledgment. This paper presents research results of the Belgian Network DYSCO (Dynamical Systems, Control, and Optimization), funded by the Interuniversity Attraction Poles Program, initiated by the Belgian State, Science Policy Office. The scientific responsibility rests with its authors. This work is also supported in part by "Communauté française de Belgique - Actions de Recherche Concertées".

Finally, the authors would like to thank S. Chafik and J.C. Delvenne for their contributions on various aspects of this work.

References

1. Alzate, C., Suykens, J.A.K.: Sparse kernel models for spectral clustering using the incomplete cholesky decomposition. In: IJCNN, pp. 3556–3563 (2008)
2. Alzoubi, H., Pan, W.D.: Fast and accurate global motion estimation algorithm using pixel subsampling. Information Sciences 178(17), 3415 (2008)
3. Beucher, S.: The watershed transformation applied to image segmentation (1991)
4. Blondel, V.D., Guillaume, J.L., Lambiotte, R., Lefebvre, E.: Fast unfolding of communities in large networks. Journal of Statistical Mechanics: Theory and Experiment 2008(10), 10008 (2008)
5. Cox, I., Rao, S., Zhong, Y.: "ratio regions": A technique for image segmentation. In: International Conference on Pattern Recognition, vol. 2, p. 557 (1996)
6. Delvenne, J.C., Yaliraki, S.N., Barahona, M.: Stability of graph communities across time scales. PNAS 107(29), 12755–12760 (2010)
7. Felzenszwalb, P.F., Huttenlocher, D.P.: Efficient graph-based image segmentation. Int. J. Comput. Vision 59, 167–181 (2004)
8. Fortunato, S.: Community detection in graphs. Physics Reports (2010)
9. Frederix, K., Barel, M.V.: Sparse spectral clustering method based on the incomplete cholesky decomposition. Report TW552, Katholieke Universiteit Leuven (2009)
10. Girvan, M., Newman, M.E.J.: Community structure in social and biological networks. Proceedings of the National Academy of Sciences of the United States of America 99(12), 7821–7826 (2002)
11. Good, B.H., de Montjoye, Y.A., Clauset, A.: Performance of modularity maximization in practical contexts. Physical Review E 81(4), 046106+ (2010)
12. Ho, J., Lee, K.C., Yang, M.H., Kriegman, D.: Visual tracking using learned linear subspaces. In: Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, vol. 1, pp. I–782–I–789 (2004)
13. Olszewska, J.I.: Unified framework for multi-feature active contour. Ph.D. thesis, Universite catholique de Louvain, Ecole polytechnique (2009)
14. Kass, M., Witkin, A., Terzopoulos, D.: Snakes: Active contour models. International Journal of Computer Vision 1(4), 321–331 (1988), doi:10.1007/BF00133570

15. Martin, D., Fowlkes, C., Tal, D., Malik, J.: A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In: Proc. 8th Int'l Conf. Computer Vision, vol. 2, pp. 416–423 (2001)
16. Newman, M.E., Girvan, M.: Finding and evaluating community structure in networks. Physical review. E, Statistical, nonlinear, and soft matter physics 69(2 Pt 2) (2004)
17. Reichardt, J., Bornholdt, S.: Statistical mechanics of community detection. Phys. Rev. E. Stat. Nonlin. Soft Matter Phys. 74(1 Pt 2) (2006)
18. Shi, J., Malik, J.: Normalized cuts and image segmentation. IEEE Transactions on Pattern Analysis and Machine Intelligence 22(8), 888–905 (2000)
19. Sundaramoorthi, G., Mennucci, A., Soatto, S., Yezzi, A.: Tracking deforming objects by filtering and prediction in the space of curves. In: CDC (December 2009)
20. Wang, S., Siskind, J.M.: Image segmentation with minimum mean cut. In: Proceedings of the Eighth IEEE International Conference on Computer Vision, ICCV 2001, vol. 1, pp. 517–524 (2001)
21. Werman, M., Peleg, S., Rosenfeld, A.: A distance metric for multidimensional histograms. Computer Vision, Graphics, and Image Processing 32(3), 328–336 (1985)
22. Wu, Z., Leahy, R.: An optimal graph theoretic approach to data clustering: theory and its application to image segmentation. IEEE Transactions on Pattern Analysis and Machine Intelligence 15(11), 1101–1113 (2002)