Segmentation of images

Laura Bravo
Universidad de los Andes
Cra 1 #18a-12, Bogotá, Colombia
lm.bravo10@uniandes.edu.co

Juan Pérez
Universidad de los Andes
Cra 1 #18a-12, Bogotá, Colombia
jc.perez13@uniandes.edu.co

Abstract

*Segmentation is one the elementary problems of computer vision and consists of partitioning an image into sections. Namely, each pixel is assigned a certain label and must share some characteristics with other pixels with the same label; the ultimate goal is that this information can be related to finding objects in the image as well as their boundaries. Here, a simple segmentation algorithm was implemented as a MATLAB function. The function can take into account color information (coming as HSV, La*b* or RGB), spatial information, if the user requests it, can be asked to perform different clustering algorithms for label assignment and, finally, will look for a user-determined amount of clusters, which should correspond to the amount of objects in the image. All the possible combinations of color model and classification algorithm where tested on a subset of the BSDS500 dataset, where results showed that the two best-performing combinations where* ***bla*** *and* ***bla****, with measures of* ***XX*** *and* ***XX*** *respectively.*

1. Introduction

Segmentation can be considered as a problem of classification, since each pixel can be given the 'class' of the object it belongs to. The pixel will be considered to belong to a determined object or region of the image if it satisfies some characteristics of the object, that is, if it is somehow similar. Similarity can also be thought of as a sort of *closeness* in some space, therefore, the problem can be broken into two subproblems: find a representation of pixel's characteristics in some space and, define a method for classifying pixels close to each other as belonging to the same class.

The first problem, a good representation space for the object, can be tackled by thinking of attributes of its pixels, such as color and, maybe, position of the pixel. Color is a powerful feature to extract from a pixel for this purposes, since objects depicted in pictures usually have a characteristic range of colors different from other objects. Also, it would be a good assumption to say that colors 'near' to each other are part of the same object. Nevertheless, one can have different objects with the same color, and if color is the only variable being taken into account, misclassification may occur. A way to handle this is to add another variable to the pixel, its spatial position. Just like with colors, it is common for pixels belonging to the same object to be spatially close to each other; however, if spatial distance is the only variable being measured, misclassification will occur, too. Accordingly, it seems a rather good idea to either select color or color *and* position as the variables representing the pixel.

Defining a method for classifying pixels close to each other as belonging to the same class can be easily treated through three algorithms: *k-means*, *gaussian mixture model* and *hierarchical clustering*. All of these need a parameter (among others), say k, that states how many clusters it will look for in the representation space. In the case of *hierarchical clustering*, this parameter is not the exact number of clusters but the maximum number of clusters it will accept. The fundamental difference among them appears between the *gaussian mixture model* and the other two, since this algorithm will provide a soft assignment to a class, which means that it will not just claim that a point in the space belongs to a given class but, rather, that it belongs to every class with a certain probability.

There is also another tool that could be used to segment the image: the watershed transform. This algorithm is not for clustering pixels according to some similarity between them, but its output generates closed regions of pixels, which is the fundamental objective of segmentation. Therefore, the watershed transform is also offered as an option in the function generated here.

In the present work the problem of segmenting an image based on the mentioned representations of the image, was tackled by varying the classification algorithms, and the color spaces. Additionally, the comparison was made between taking and not taking into account the position of the pixels in the image.

2. Materials and methods

A MATLAB function for performing segmentation over an image was implemented. This function takes four inputs: (i) the image in *rgb* space; (ii) the feature space, which is the way the user wants the algorithm to represent each pixel, the available options are *'rgb'*, *'lab'*, *'hsv'*, *'rgb+xy'*, *'lab+xy'* and *'hsv+xy'*, needless to say, the options that have *'+xy'* are the ones that take into account the spatial position of the pixel; (iii) the clustering method the algorithm will use, which can either be *k-means*, *gaussian mixture model*, *hierarchical clustering* or *watersheds*; and (iv) the number of clusters to be found in the image, which should correspond to the number of objects present.

Within the function, there was a discrepancy regarding the scale (range) of each space. This problems arose because channels may not be comparable amongst themselves, like channel *h* and channel *v* of *hsv*. But more importantly, because distances are not comparable between color spaces, since they depend on the size and scale of the image and the definition of the color space. This was solved by normalizing each channel, that is, finding the *zscore* of each of the computed channels (here spatial location is considered to be a channel, too, if the user requested it to be taken into account), so that only the distance of the point to the origin of its dimension is the relevant information.

2.1. Classification algorithms

As previously mentioned, four different classification algorithms were used in the function, so that the user could have control over it. Next, a brief description of each one of them will be given.

2.1.1 k-means

K-means clustering is a procedure that allows solving of an optimization problem, like minimizing the distances from points in space to given centroids, through a finite amount of iterations in which the centroids are computed again and again based on the data points. This algorithm requires two important things as initialization parameters, the number of clusters it will look for, k, and the initial centroids, which are usually randomized [1, 2]. The procedure is as follows: (i) Initialize k centroids, (ii) assign labels to each point in the space according to the centroid that is closest to it, so that k clusters are formed, (iii) compute the center of mass for each of the clusters and define these centers of mass as the new centroids, (iv) repeat steps *ii* and *iii* until convergence of the assignations.

This procedure has three main disadvantages. First, one has to set k, the number of clusters that are to be found, which is not clear from the data itself; therefore, this will always need a human agent stating this. Second, the iterative assignation of labels requires a large number of computations, which is very expensive in terms of time. Third, the solution to the optimization problem depends highly on the initialization and, accordingly, it could converge to a local minimum.

2.1.2 Gaussian mixture model

A generalization of *k-means* is the mixture of Gaussians. Here the idea is to represent each group with a Gaussian distribution, so that, like the iterative computation of the centroids in *k-means*, the problem is reduced to optimally fit the model to the data through the estimation of its parameters. The model makes the assumption that the data that was taken comes from a normal distribution and, consequently, at each iteration the mean of each distribution and its covariance matrix must the estimated. Similarly to *k-means*, this model also requires knowing how many distributions are to be found as well as a random initialization of parameters. A fundamental difference between this model and *k-means* is that the assignment of labels, which is stating what distribution does a point belong to, is not 'hard' but rather 'soft', so that every point belongs to every distribution but with some *responsibility*. The algorithm is very similar to *k-means* and proceeds as follows [2]:

I don't like the fact that this is in a list.

1. Initialize k normal multivariate distributions.
2. Estimate responsibilities of each point in the space according to the current distributions.
3. Compute new parameters for each of the distributions given the previously estimated responsibilities.
4. Repeat steps 2 and 3 until convergence of the responsibilities.

This algorithm presents improvements with respect to *k-means*, but still suffers from some of the same problems. For instance, it takes into account that the distance to the center of all the distributions may not be comparable, since it depends on the 'elongation' of the distribution in that direction; additionally, it allows the previously stated soft assignments. However, the number of computations increases and, like in *k-means*, the k parameter have to be known and the initialization may cause the finding of a local minimum.

2.1.3 Hierarchical clustering

This algorithm takes a distance metric between clusters and creates a hierarchy of partitions of the space based on it. The clusters are constructed in an agglomerative fashion by iteratively increasing a threshold distance to which clusters are considered to be one and, therefore, are merged. An interesting property of this procedure is that one can pick a

distance metric that takes into account some characteristics of the problem that is being studied, which makes it more flexible; nevertheless, it is a very slow algorithm for problems with large amount of data, since a lot of distances have to be computed iteratively [2].

2.1.4 Watershed

The watershed transform is based on the interpretation of a one-dimensional image as a topological surface. The idea is to divide the image into sections where water would rest, that is, areas drained by different river systems [3]. It starts by making holes on the lowest parts of the images and then flooding the image as if the topological surface could be sunk in a bathtub; then, gradually, individual 'lakes' will touch and that is where a division is drawn, creating a segmentation effect. Usually, the watershed transform does not perform well on itself, since there is a large number of areas in the image that meet these characteristics, therefore, it is common to use techniques of minima-imposing based on some markers, that can be generated trough the extended-minima transform [2].

2.2. Evaluation of the algorithm

For the evaluation, the segmentations product of all the combinations of clustering methods and color space options were quantitatively compared with the ground truth. In total, by applying all the methods and color spaces to 3 images, the evaluation produces in total 72 segmentations. Because there were multiple annotations for each test image in the **BSDS name?** dataset, the evaluation depended on choosing the best of these per image. For any given segmentation, the Jaccard index (intersection of the segmentation and annotation, over the union of the two) was calculated for the 3 largest objects in the ground truth. This was then averaged to attain one value for the whole segmentation. Due to the fact that an object in the ground truth can correspond to more than one in the segmentation, there was a selection process where the largest element of the segmentation area, corresponding to the annotation object, was taken. It is important to point out that the parameter of number of clusters required in the segmentations was made to match the number of groups in the ground truth, but later on it was modified to be a fixed number.

3. Results

Table 1 depicts the results of the evaluation averaged by the images used, that is there is one value per clustering method per color space option. The average by method implies that the best method in all the color space options was GMM, scoring consistent values, a trend that is maintained in all the other methods. The lack of variance in the watershed method suggests that the algorithm fails to draw significant information from the color options. Nevertheless, the color space options did affect the performance of the remaining clustering methods. For each one of them, there was a color space in which they performed better, for this reason this parameter must be fine tuned when selecting a unique clustering method for image segmentation.

Table 1. Average Jaccard Index. Variable number of clusters subject to the ground truth

	Method			
	K-means	GMM	Hierarchical	Watershed
RGB	0.32	0.53	0.27	0.48
Lab	0.30	0.49	0.34	0.48
HSV	0.31	0.51	0.32	0.48
RGB + xy	0.35	0.51	0.39	0.48
Lab + xy	0.36	0.51	0.38	0.48
HSV + xy	0.35	0.45	0.37	0.48
Average	**0.33**	**0.50**	**0.35**	**0.48**

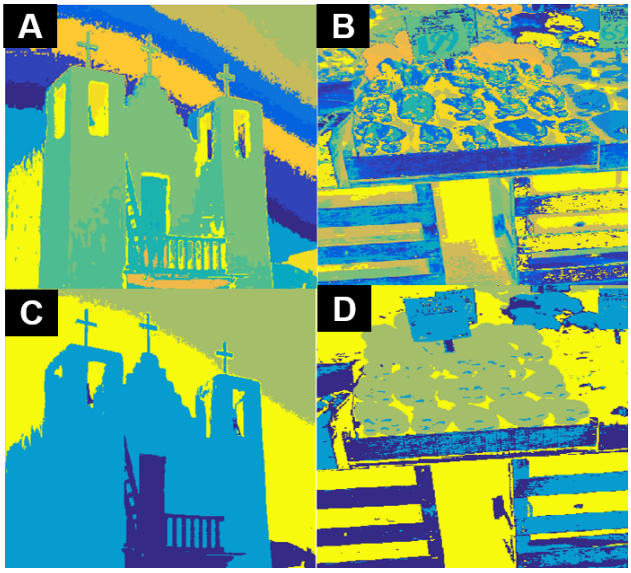

Figure 1. Examples of over-segmentation. The segmentations A) and C) were done in the Lab space. B) and D) are in the RGB space. All of them were obtained with hierarchical clustering. A) and B) correspond to variable clusters and C) and D) to 4 clusters.

All of the methods across all the color spaces have a Jaccard Index that does not surpass significantly the half point of the accuracy. This could be due to the selection of the number of clusters not being the most appropriate for all the images. Because of the representation space (color and space), the use of an elevated number of clusters created over-segmentation in the images, which in turn leads to a diminishing in the Jaccard Index overall. In Figure 1 there is an example of this over-segmentation experienced with the variable number of clusters, and the additional improvement seen by fixing them to a smaller number. To alleviate

this problem in the future it is recommended to optimize said parameter.

In order to reduce the computation time, and evaluate the effect of the parameter number of clusters in the segmentations, said parameter was fixed to 4 clusters for all the methods. The results are found in the Table 2. In general, with the decrease in the number of clusters, there was an increase in the average Jaccard Index per method. Nevertheless, the best method is still GMM, but in this case, the color option had a significant effect in the performance of the segmentation. Ultimately, GMM was most improved by using the Lab space and the x, y coordinates of each pixel. These results confirm the need for adjusting properly the parameter of the number of clusters. Additionally, it is important to take into account the effect of the objects evaluated from the annotations and their relation with the previous parameter. Also, because the evaluation strategy depends upon taking one of the 5 available annotations, there exists an augmented risk of creating a bias that may favor inequitably certain methods over others.

Table 2. Average Jaccard Index. Fixed number of clusters: 4

	Method			
	K-means	GMM	Hierarchical	Watershed
RGB	0.45	0.53	0.48	0.30
Lab	0.44	0.55	0.45	0.30
HSV	0.37	0.57	0.41	0.30
RGB + xy	0.33	0.52	0.48	0.30
Lab + xy	0.39	0.66	0.37	0.30
HSV + xy	0.40	0.42	0.41	0.30
Average	**0.40**	**0.54**	**0.44**	**0.30**

A common source of error for the algorithm is the effect of texture in the images. Figure 2 exemplifies this. All of the panels correspond to the segmentation in the same color space HSV+xy, but each of them was done by a different clustering method. The over-segmentation in these images is due to the added detail that the texture of the water adds. The representation spaces available, that is, color and spatial coordinates are not enough to explain the images sufficiently and thus the segmentation does not as of yet correspond fully to the annotations. This limitation could be addressed by adding an additional dimension to the representation space that includes texture information. Another way to do this could be by preprocessing the images with filters so as to eliminate details.

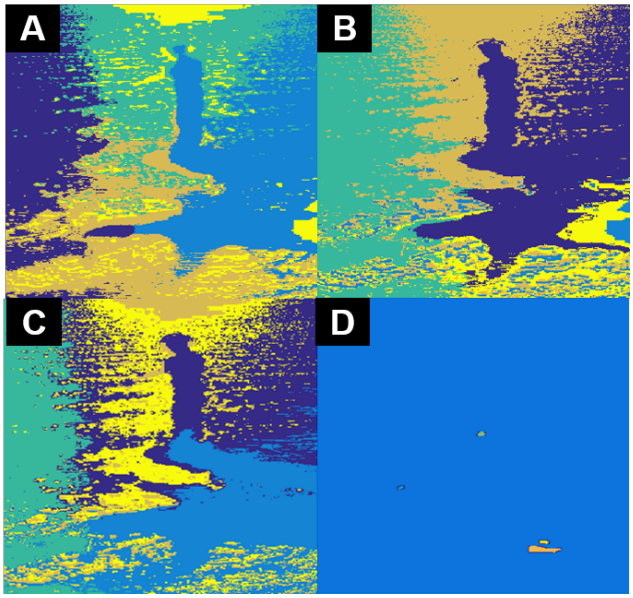

Figure 2. Examples of an image with significant texture. All the segmentations were done in the HSV+xy space, the methods here are A) k-means, B) GMM, C) Hierarchical and D) Watershed

References

[1] E. Alpaydin, Introduction to Machine Learning, 2nd ed. London, England: The MIT Press, 2010.

[2] P. Arbeláez, Lecture 5: Clustering, Computer Vision, Universidad de los Andes, Colombia, 2017.

[3] "The Watershed Transform: Strategies for Image Segmentation - MATLAB & Simulink", Mathworks.com, 2017. [Online]. Available: https://www.mathworks.com/company/newsletters/articles/the-watershed-transform-strategies-for-image-segmentation.html. [Accessed: 13- Mar- 2017].