Application of the Wrapper Framework for Image Object Detection

Michael Farmer
Computer Science, Engineering Science & Physics
University of Michigan at Flint, MI, USA
farmerme@umflint.edu

Abstract

Tools for automatic image understanding for managing operator workloads are essential. One common task for image analysts is the scanning large collections of real-time images looking for particular objects of interest. This task is difficult to automate due to variable imaging geometries and environmental conditions. This variability of conditions can make automating image strong segmentation for eventual object classification extremely difficult. This paper proposes a tool which integrates image segmentation and classification to allow the integration of semantically meaningful information into the segmentation process. The wrapper framework has previously been shown to be effective in performing strong segmentation on images containing large complex shapes in a fixed field of view. This research extends the applicability of wrapper to wide area surveillance of images containing possibly multiple objects of interest. The approach is demonstrated on aerial images from the Katrina disaster to be able to detect buildings for possible damage assessment.

1. Introduction

Traditional methods for object detection in images involve either matched filter detectors which are convolved over the image, or else strong image segmentation followed by classification of the resultant segmented regions in the image. Neither of these have lived up to their potential due to (i) the inflexibility of the first approach in detecting objects of varying scale and orientation in varying collection conditions, and (ii) the inherent semantic gap between these segmentation and classification in the second approach. Existing segmentation algorithms are built upon the following two common underlying assumptions; (i) the object homogeneity with respect to some characteristic, and (ii) difference between adjacent regions. In this paper, we propose improving the segmentation process by infusing semantic knowledge into the segmentation process by combining the problems of segmentation and classification through a wrapper framework. In [6] Li et al. have noted, "it is often difficult....to determine which regions....should be used for the final segmentation". The goal of the wrapper framework is to directly address this problem by integrating segmentation processing with a classification process to provide the required semantic information.

The tools available for classifying an object for image retrieval include texture, color, shape and structure. Since texture and color are used as low-level cues, shape and structure are the remaining features. The use of structure has been an active area of research with good results when applied to relatively well-formed images with relatively simple backgrounds, such as those for horse recognition by Cours and Shi [7]. In their approach regions in the image are initially associated with key structures of the object of interest, and then a novel algorithm is used to combine the regions to best match the desired object. The approach is based on a convolution of the template across the image to find objects of interest, which, when the object of interest is quite small relative to the image field-of-view, can be quite time consuming. Borenstein and and Ullman and then Borenstein and Malik also propose an approach where they used low level shape fragments and built an object of interest based these. They use a shape template of object consisting of structurally significant object components (e.g. limbs, head, etc) to guide the assembly of the fragments [9]. In order to not require mapping of image regions with sub-structures we adopted shape as the mechanism for inserting semantic information into the segmentation process. Li et al. have a similar approach to ours of assembling sub-regions to derive the final segmentation however; their approach compares not only foreground, but also background regions in the two images which can be particularly limiting if the object of interest is considerably smaller than the field of view of the imaging device such as in a surveillance application. The proposed approach is demonstrated on a collection of aerial images from the Katrina hurricane disaster in Louisiana, and an example image can be seen in Figure 1.

2. Description of system

The wrapper framework for integrating the classification and segmentation tasks was motivated by the use of wrapper methods for pattern classification feature selection, where features in a pattern classification problem are selected by testing the effect their inclusion would have on the final system classifier performance, and has been shown to be effective for a variety of applications including automotive airbag suppression and mammographic tumor detection [1][2]. The wrapper framework for feature selection wraps the selection process inside the classifier. In an analogous manner the wrapper-based segmentation system wraps the

978-1-4244-2175-6/08/$25.00 ©2008 IEEE

region selection for strong segmentation inside the classifier. Thus in-depth semantic knowledge can be integrated into the segmentation process via the classifier and its application of a pre-stored database of exemplar objects of interest.

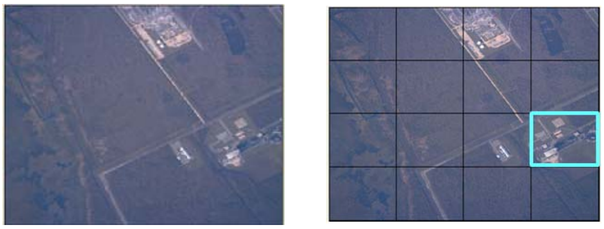


Figure 1: Example image: (a) Original image from Katrina disaster, and (b) mosaic of original image.

The general processing flow for the entire wrapper framework can be seen in Figure 2. The processing is divided two distinct phases, (i) conventional (context-free) segmentation, and (ii) wrapper-based (semantic) segmentation. In the conventional segmentation phase no contextual or semantic information is used and the image is segmented based on a low-level homogeneity metric for the pixels. In this application, color is used as the homogeneity metric, however, grayscale has been used in past wrapper applications (see [1][2]), and texture is also applicable. The second phase is the wrapper segmentation phase where the critical semantic information is integrated via the classifier.

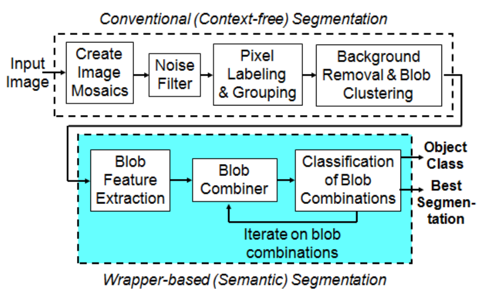
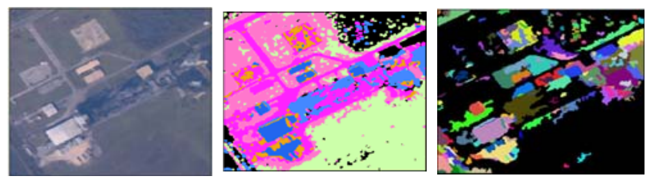


Figure 2: Processing flow for wrapper-based image segmentation.

2.1. Conventional Segmentation Processing

The conventional segmentation processing flow begins with a low level order statistic filter such as a median filter for removing high frequency image speckle. The image is then divided into a set of mosaics of typically 4x4 sub-images to limit the size of the images processed at any time for performance purposes. Figure 3 (a) shows the image from Figure 1 divided into a 4x4 mosaic of sub-images, and Figure 3 (b) shows the sub-image for the 3rd row and 4th column mosaic. In the most general case, the tool would automatically select the number of mosaics, and then process through all of them based on an optimal size of the mosaics to maximize processor efficiency, and easily support parallel implementation.

The pixel labeling and grouping stage performs low-level weak segmentation that converts the pixel values into a label, and then groups these labeled pixels into blobs. The subtasks of the processing for grouping pixels into blobs are provided in Figure 4. The Compute Data Clusters task is performed using the Expectation-Maximization algorithm. Since the wrapper framework relies on the image being over-segmented, the number of mixtures used in the EM algorithm is relatively high, with 7 mixture components being used in this study. The Label Pixels task then uses the mixtures defined by EM to label each pixel with its appropriate mixture membership, with typical results being shown in Figure 3 (a). This labeled image is then mode filtered to further remove isolated pixels.



(a) (b) (c)

Figure 3: Image mosaic results: (a) image mosaic (3, 4) from Figure 1, (b) labeled image after mode filter and small blob removal, and (c) labeled image after background removal.

In the task of Region Extraction the pixels are grouped together into blobs, or regions of common labeling based on an 8-way connected components algorithm. The system then removes blobs that fall below a user-defined threshold, with the threshold for this application being set to 50 pixels in size. At this point the image has been completely divided into regions of low-level grayscale or color homogeneity. An adjacency graph is then constructed to define the connectedness of the blobs in the image, which will be used later for cluster detection.

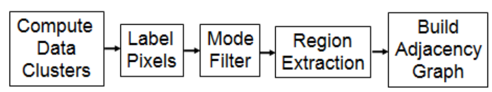


Figure 4: Processing flow for Group Pixels into Blobs.

Recall from Figure 2, the next step is Background Removal and Blob Clustering. The background in an image is defined as the larger regions that typically are not of interest. The size of the background regions is independently defined by two characteristics, the area and the length. Thus regions of large area or large regional extent (such as roads and rivers) can be ignored. The user can select the size of the regions to ignore as background, with the area threshold currently set to 1500 pixels and the length threshold set to 75 pixels. Future work will employ an automatic tool that determines the cut-off of which blob sizes to keep based on knowledge of the image collection geometry, image field of view, etc. Removing of the background, as shown in Figure 3 (b), allows the algorithm to now focus on more interesting regions, in a similar manner to human perception where background regions of common grayscale or texture are easily ignored while more interesting objects are sought. Once the background is removed, regions of clusters of blobs can readily be detected using the adjacency graph. Clusters are defined by collections of blobs that are adjacent to each other, as can be seen in Figure 3 (b). The wrapper segmentation processing then analyzes each of these blob clusters to determine if any objects if interest may be present. The ability of the wrapper framework to process clusters of blobs, rather than all the blobs in an image, is critical for performance, since the number of possible blob groupings rapidly suffers combinatorial explosion.

2.2. Wrapper Segmentation Processing

The wrapper framework processes each cluster of blobs independently and only tests combinations of the blobs within each of these clusters, thereby significantly reducing the combinatorial explosion. Clusters consisting of individual blobs are tested first against a training database to see if any of them may match an object of interest since they require no combining. Then the remaining more complex clusters are processed, where a variety of combinations of blobs are attempted to see if any of these combinations may match any objects in the training database.

Recall from Figure 2, the first task in the wrapper processing is the feature extraction for each blob. The approach taken by the wrapper framework is to use the shape information of the object of interest to provide the contextual information required to support the robust segmentation. The wrapper framework uses moments to capture the shape information of the objects of interest. The wrapper framework initially computes the desired moments order for each blob in the image and stores these values away for the subsequent classification stage. A segmented image $I(i; j)$, can be represented as the sum of its blobs, resulting in an interesting in the ability to combine the moments of blobs directly rather than combining blobs and then computing their moments [1]:

$$I(i,j)=\sum_{k=1}^{K} I^{(k)}(i,j) \rightarrow M_{lm}=\sum_{k=1}^{K} M_{lm}^{(k)} \qquad (1)$$

where $I^{(k)}(i,j)$ is the portion of the image corresponding to blob k, K is the total blobs in the labeled image, M_{lm} is the geometric moment of order *(l+m), and* $M_{lm}^{(k)}$ is the geometric moment of order *(l+m)* for blob k . Thus, we see that the moments for the entire image are the sum of the moments computed for each blob. Equation (1) allows the pre-computation of the moments for each blob, which can then be added together to compute the moments for any blob combination. This result is critical since it ensures the wrapper framework is computationally tractable.

For Blob Combiner processing, an algorithm is required which will combine subsets of the blobs in the image together while assuming a particular object class is present in the image. The simplest algorithm is to perform an exhaustive search through all the possible blob combinations. For smaller clusters consisting of a user-determined threshold (<5 for this work) the wrapper framework performs an exhaustive search. For larger clusters a more sophisticated approach must be used, since for clusters consisting of as few as 20 blobs a brute force search of every possible combination would require roughly one million combinations, and if the number of blobs only increased modestly to 25, the number of combinations would exceed 33 million. The algorithm used is derived from feature selection and is the *Plus-L, Minus-R* wrapper-based algorithm, where on each iteration L blobs are attempted to be added and then R blobs are attempted to be removed to try to improve the classification of the blob combination [1].

The blob combiner first assumes the image is from a particular class. In this application the desired class is defined by the human operator which has been tasked with searching for a particular object of interest. The combiner then iteratively generates blob combinations. For each blob combination, the moments for each blob are added together, and then this resultant moment collection defines the moments of the current blob combination. These moments are then converted to central moments, and then further converted to a desired invariant moment. For this application, the moments are made scale invariant.

For each blob combination, these moments of the corresponding segmentation are fed to the final stage of the wrapper processing.

The Classification Processing employs the nearest neighbor classifier due to its simplicity of implementation and ease of traceability, since it allows the user to see exactly which training samples provide the lowest classification distances. The moments are normalized to a range of zero to one prior to classification in order to reduce the increasingly dominant effect of the higher order moments. The value for the best classification distance $d_{class}(\{X_k\}|C)$ for each candidate class, C, is retained as well as the resultant blob combination. The pattern class C that minimizes $d_{class}(\{X_k\}|C)$ across the entire history of the execution is then considered to be the classification of the object, and the set of blobs that correspond to that classification $\{X_k\}$, is the resultant best segmentation. A subset of the training database used for this application provided in Figure 5. The training set was derived from a collection of images containing buildings that were not associated with the Katrina test images.

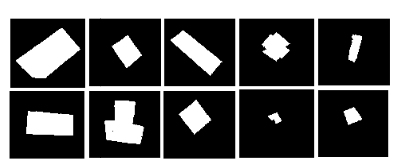
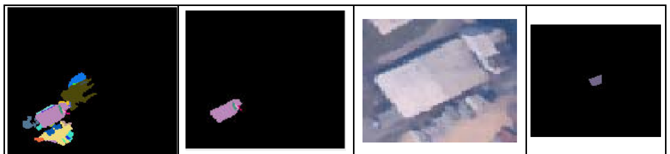


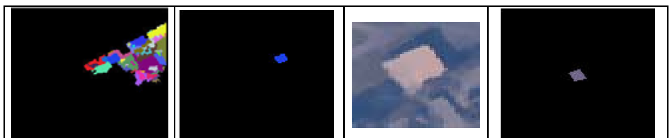
Figure 5: Subset of training set for disaster monitoring application.

Figure 6 through Figure 8 show the results of the processing on three of the clusters detected from Figure 3 (c), where in all cases a building was detected as the top choices in the region. Note the complexity of the clusters from which the wrapper algorithm is able to detect combinations of blobs are construct objects of interest based on the training samples provided in Figure 5. In particular, the cluster shown in Figure 7 consists of 45 blobs and the cluster in Figure 6 consists of 22 blobs. In all cases the algorithm began with testing all pair-wise selection of blobs, and then continued with the *Plus-L, Minus-R* processing until the algorithm converged. For structures such as buildings, there may be other man-made constructs that can match the rectangle-based framework that is common for buildings as shwon in Figure 9



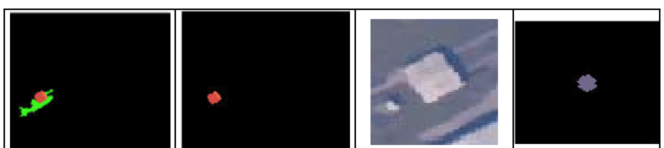
(a)	(b)	(c)	(d)

Figure 6: Best match for cluster: (a) Original blob cluster, (b) final blob combination, (c) image region and (d) best training sample.



(a)	(b)	(c)	(d)

Figure 7: Best match for cluster: (a) Original blob cluster, (b) final blob combination, (c) image region and (d) best training sample.



(a)	(b)	(c)	(d)

Figure 8: Best match for cluster: (a) Original blob cluster, (b) final blob combination, (c) image region and (d) best training sample.

Table 1 provides the results for processing on the mosaic *(3, 4)* shown in Figure 3. Note the system was able to successfully detect all of the buildings in the mosaic and only wrongly detected two image regions as buildings that were actually background, however, as can be seen in Figure 9 there were man-made constructs whose shape resembled a building. Overall the use of the wrapper framework as an image pre-processor to support image analysts and reduce their workload appears promising, as the operator is referred to regions of the image that may require further detailed analysis without the operator having to scan the image.

(a)	(b)	(c)	(d)

Figure 9: Best match for cluster: (a) Original blob cluster, (b) final blob combination, (c) image region and (d) best training sample.

Table 1: Wrapper Results on Image in Figure 3.

	Blob Combinations Detected as Objects	Blob Combinations Detected as non-Objects
Actual Objects	5	0
Actual Clusters not Containing Object	2	7

3. Conclusions & future work

As seen in the paper, strong image segmentation requires infusing semantic knowledge into the segmentation process. We demonstrate the use of the wrapper framework as a vehicle for combining the problems of segmentation and classification. By providing semantically meaningful information to the segmentation algorithm we are in a better position to segment an object despite internal color, grayscale or texture variations. Since the combinatorial explosion is still a potential issue with large blobs, future research will be directed at developing a sub-graph based *Plus-L Minus-R* algorithm that only considers blob combinations that are contiguous, which greatly reduces the number of possible blob combinations. Likewise we are currently developing a complete image query system based on the wrapper framework to support the search through an image database of user defined shapes.

References

[1] M. Farmer and A. Jain, "Wrapped based approach to image segmentation and classification", *IEEE Transactions in Image Processing*, vol. 14 no.12, pp 2060-2072, 2005.

[2] H. Rabiei, A. Mahloojifar and M. Farmer, "Providing context for tumor recognition using the wrapper framework", *IEEE International Symposium on Biomedical Imaging*, 2007.

[3] http://ngs.woc.noaa.gov/katrina/KATRINA0000.HTM.

[4] J. Flusser and T. Suk, "On the calculation of image moments", *Research Report #1946, Institute for Information Theory and Automation, Academy of Sciences of the Czech Republic*, 1999.

[5] J. Luo and C. Guo, "Perceptual grouping of segmented regions in color images", *Pattern Recognition*, vol. 36, pp. 2781-2792, 2003.

[6] J. Li, J.Z. Wang, and G. Wiederhold, "IRM: Integrated region matching for image retrieval", *Proc. 8th ACM Intl Conf. on Multimedia*, pp. 147-156, 2000

[7] T. Cour and J. Shi, "Recognizing objects by piecing together the segmentation puzzle", *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, pp. 1-8, 2007.

[8] M. Safar, C. Shababi, and X. Sun, "Image retrieval by shape: A comparative study", *Proc. IEEE International Conference on Multimedia and Exposition*, pp. 141-144, 2000.

[9] E. Borenstein and J. Malik, "Shape Guided Object Segmentation", *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, pp. 969 – 976, 2006.

[10] E. Borenstein and S. Ullman, "Class-Specific, Top-Down Segmentation", Lecture Notes in Computer Science, vol. 2351, Springer-Verlag, pp. 109-122, 2002.