2006 IEEE International Conference on
Multisensor Fusion and Integration for Intelligent Systems
September 3-6, 2006, Heidelberg, Germany

Neural Network Based Foreground Segmentation with an Application to Multi-Sensor 3D Modeling

Miti Ruchanurucks, Koichi Ogawara, and Katsushi Ikeuchi

Abstract - **This paper presents a technique for foreground/background segmentation using either color images or a combination of color and range images. In the case of images captured from a single 2D camera, a hybrid experience-based foreground segmentation technique is developed using a neural network and graph cut paradigm. This gives an advantage over methods that are based on color distribution or gradient information if the foreground/background color distributions are not well separated or the boundary is not clear. The system can segment images more effectively than the latest technology of graph cut, even if the foreground is very similar to the background. It also shows how to use the method for multi-sensor based 3D modeling by segmenting the foreground of each viewpoint in order to generate 3D models.**

Index Terms - Interactive image segmentation, Machine learning, Multi-sensor 3D modeling.

I. INTRODUCTION

ALTHOUGH foreground segmentation plays an important role in many areas, not only in editing images or video, but also in enhancing medical images or extracting a boundary before performing object recognition, the segmentation problem is still imperfectly solved.

Recent approaches of 2D foreground/background segmentation attempt to extract the foreground using color and/or contour information. Most such methods require human input: a user marks the areas as foreground/background or marks a contour for use as a guideline.

An example of a method that exploits color is Bayes matting [1], which relies on color distribution. A user marks foreground/background roughly, then the "alpha" value - the method's descriptor of being foreground/background - is computed over the remaining region. The method is fairly successful, but one drawback is that color distributions must be sufficiently separated between foreground/background.

In a method that exploits edge information, such as "intelligent scissors" [2], an image's contours must be clear. When a contour contains too much texture, the system is not able to choose between multiple edges. Conversely, if an image's edges are too smooth, the method likewise suffers because contours cannot be found.

In light of these limitations, combining both color and edge information seems to offer the most promise. Graph cut [3], [4] is one such powerful technique. The optimal contour can be found based on an objective function which in general minimizes the composition of a term that represents the fitting between the estimated alpha value vs. actual color and another term that represents smoothness. However, even the most powerful graph cut method [5] still suffers if the foreground/background colors are too ambiguous.

Meanwhile, in the field of object recognition, color distribution based methods are of limited use, as color alone is usually inadequate to represent patterns. Instead, machine learning dominates this field. [6] focuses on combining belief and neural networks to apply a set of labels (e.g. sky, vegetation) to images. [7] uses a probability model to extract an object of interest based on input training data without human interaction. This method also exploits graph cut during the segmentation phase. These techniques, however, focus on unsupervised object recognition more than on how to effectively segment the foreground.

Supervised learning techniques such as [8] detect meaningful edges in images by exploiting brightness, color and texture. They compare the use of density estimation, decision trees, logistic regression, hierarchical mixtures of expert systems, and support vector machines. However, this method cannot extract the foreground directly. [9] also compares various machine-learning algorithms such as neural network, nearest-neighbor, and decision trees, with the objective being to recognize a hand in moving images. The result is good even in a changing environment; however, a lot of user interaction is required during the initial setup.

Incorporating the advantages of both segmentation methods, our system augments the decision-making criterion of graph cut with a supervised machine-learning algorithm. In addition to each pixel's characteristics being used as training data, gradient information is exploited. Furthermore, the machine-learning algorithm is self-learning until the contour between foreground/background is located.

For multi-dimensional images, the method described in [10] shows how to apply segmentation on stereo images and 3D images, like medical images. Whereas [11] divides video object segmentation into a model detection and a tracking problem, [12] and [13] consider videos as 3D objects and use graph cut to segment the foreground instead. [14] uses special video sensors to capture and extract the foreground automatically.

Another potential application of foreground segmentation on multidimensional image is 3D object modeling. Generally,

to model an object, pairs of color and range images taken from different views can be used. Then, before generating the virtual model of the foreground, users must delete the background manually, which is a tedious task. It will be shown that our segmentation algorithm can be directly used to extract the foreground of each view before composing the virtual object.

The key problem, segmentation, is what our method directly addresses. In the next section, foreground segmentation using a neural network is explained. Section 3 explains how to use the method to do segmentation using multi-sensor modeling. Finally, Section 4 contains our conclusions.

II. Hybrid Segmentation

As mentioned earlier, a method that exploits both color and edge information is preferable, such as graph cut. Since the original graph cut algorithm was developed [4], there have been various improvements. [3] focuses on a multidimensional graph cut. [15] improves the graph cut method by allowing interactive estimation and incomplete labeling, both of which reduce the amount of needed user interaction. However, this method's border matting technique can be applied only when an image's boundaries are smooth. [16] incorporates interactive boundary editing by representing the border as a set of vertices. Users can drag a vertex to adjust the shape of the foreground; unfortunately, this technique cannot deal with thin and branch structures well. [5] solves the border problem of [15] automatically by using belief propagation for matte estimation. The result is quite clean even when the boundary between foreground/background is not smooth. However, this system can take up to 20 minutes to process an input image of 640*480 pixels. Furthermore, the authors note that their method shares a similar weakness to Bayes matting for images in which the foreground/background colors are too ambiguous. Also in [15]; for the admittedly most difficult-to-segment image mentioned in the paper, a Gaussian Mixture Model (GMM) after convergence is well separated. This indicates the drawback of such probabilistic models in the case of color ambiguity.

In the field of pattern recognition, many powerful tools are available. Recently, Support Vector Machines (SVM) are among the most active topics in the classification field due to their simplicity and fast behavior in learning new input. In this method, the input domain is transformed to a more easily classified domain using a transforming function. Then, in the transformed domain, the best vector that can divide two regions is calculated. Unfortunately, this simplicity is actually a drawback of SVM, because there is no guarantee that the transformed domain offers such a separation. Until now, a neural network seems to be the most useful tool. It can deal with high dimensional data without assumptions about the parametric relationships between them. And, the number of patterns present in images can be approximated to adjust the network structure.

In work that focuses on exploiting patterns in the image, [9] uses color in RGB domains, hue, and saturation of a large window as the input parameter set of a machine-learning algorithm to distinguish hands in moving images. Interestingly, the method uses 175 input parameters per pixel. So, although the paper asserts the superiority of a neural network, with such large input dimensions one is forced to use a simpler technique, and user interaction is required to specify foreground/background training data. If this is done, the machine can successfully return a segmented image; the user then marks the true negative and false positive areas for use as new training data. This iterative process continues until the user is satisfied with the recognition result.

One might wonder why there are such large differences between foreground/background segmentation and object recognition algorithms when the objective, segmentation, is the same. This question can be explored by comparing the two most successful current methods, [5] and [9]. While the former is successful in the sense that little user interaction is needed to segment an image, it cannot deal with the case in which foreground/background colors are too ambiguous. On the other hand, the latter method exploits a large number of patterns to generate a robust classifier, but a lot of user interaction is required, and smoothness in boundaries is not guaranteed. Also, when foreground/background patterns are too similar, machine learning has difficulty learning how to classify them. Therefore as described below, our method attempts to exploit the strengths of both these methods while overcoming their weaknesses.

A.Hybrid Method

Based on its main strength - effectiveness in recognition – a neural network is the basis of this work. Although [9] is successful in using high dimensional training input which forces the use of a comparatively fast but weak classification method, we have found that this larger window does not always produce good recognition results. Especially when the foreground/background consists of many different regions, using a larger window usually confuses the classifier because it encounters patterns which are too complex.

Therefore, in our method, the RGB, hue, and saturation data of a pixel are used. To tackle the problem of color ambiguity, a parameter that represents texture around the pixel is computed. In characterizing texture, we compared the use of variance, entropy, and orientation of a 5x5 window around the pixel and found that variance outperforms the other parameters. Also, working with RGB or gray-scale spaces does not result in a substantially different recognition result, so

the variance of a 5x5 gray-scale image is used. Though the parameters used are similar to ones used in a supervised learning method of [8], they focus on detecting meaningful edges in the image instead of the boundary of the foreground.

At present, a network with one hidden layer is used. The number of hidden nodes is arbitrarily set to 30, which should be adequate to deal with segmentation. A larger number of hidden nodes are of course possible; the tradeoff is that of computation time.

For the transfer function of the neural network, a sigmoid function is used so that the output range is between 0 (background) and 1 (foreground):

$$out\_node = \frac{1}{1+\exp(-in\_node)} \quad (1)$$

If only this neural network is used, the result is noisy, as expected, as shown in Fig. 1. But rather than having a user mark false positives and true negatives as in [9], a self-learning paradigm can be exploited.

(a) (b)

Fig. 1: Neural network. (a) User-marked data. (b) Recognition result

First, the neural network, trained from user-marked data, is allowed to search for foreground/background. Searching employs a moving 3x3 window and starts from the green/red lines marked. Preparing for self-learning, a best candidate for training is picked up by a 3x3 window while others are judged to be foreground/background.

The search range is controlled by gradient information; high gradient indicates that the pixel is likely part of an edge, so the threshold to judge that it is foreground/background should be high:

$$pixel = \begin{cases} fore; \alpha \geq 0.5 + th\_NN \\ back; \alpha \leq 0.5 - th\_NN \end{cases} \quad (2)$$

where α is the alpha value calculated from the neural network, and the threshold th_NN is calculated using the sigmoid function:

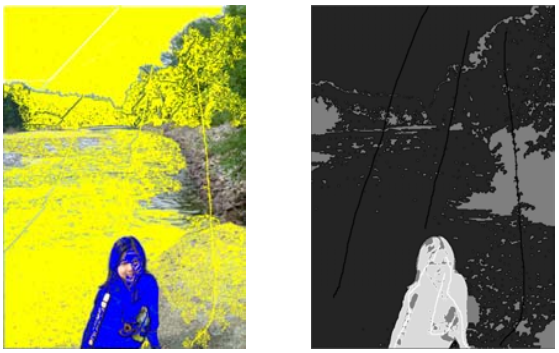
$$th\_nn = \max\left(\frac{1}{1+\exp(-grad)*k\_sigmoid} - 0.5, 0\right) \quad (3)$$

where $k\_sigmoid$ which controls the position of the sigmoid in the gradient domain is iteratively computed until the foreground and background do not touch each other, at the end of the first iteration. In this method the sigmoid forces the process of cutting to stop at the boundary more accurately than compared to using an exponential function when calculating the quality of the boundary, as is [5].

Each iteration stops when the foreground and background regions cannot grow anymore. Then the neural network is retrained using the candidate pixels sampled region-wise, as shown in Fig. 2. The candidate pixels comprise:

1) *Newly recognized candidate pixels*
2) *The training data of former iterations*
3) *(Initially) user marked data that contain the yet-to-be-recognized surrounding pixels.*

The purpose of this strategy is threefold: the neural network must be based on 1) new data, in order to become smarter, 2) its experience during former iterations, and 3) any foreground/background divisions made by a user.



(a) (b)

Fig. 2: First iteration of searching. (a) Selecting candidate pixels from foreground (blue)/background (yellow). (b) The recognized foreground (light gray)/background (dark gray). The unrecognizable part is shown in intermediate gray.

Furthermore, after the first iteration, the area left unchecked is around the boundary; note that that the criterion of (2) would not yield good results if the boundary is smooth. Using a criterion similar to [5], (2) is replaced by the new criterion:

$$pixel = \begin{cases} fore; \alpha \geq 0.5 + th\_NN \;\&\; \alpha \geq \alpha 0 \\ back; \alpha \leq 0.5 - th\_NN \;\&\; \alpha \leq \alpha 0 \end{cases} \quad (4)$$

where $\alpha 0$ is an alpha value of a seed foreground/background pixel.

Also, the criterion of color similarity is used to improve efficiency, similar to [16]. The original color image is segmented into small regions based on color similarity. Then region growing, from seed foreground/background pixels, is used along with (4) to assign new foreground/background pixels.

The process of learning continues until either the number of

candidate foreground or background pixels is considered to be insufficient to further train the network. Here, a typical rule of thumb is to use 10% of the image size. If the number of candidate data is below the above criterion, the process of learning stops. After this, instead of using a method of [15], in which a region might erode or dilate, we dilate using gradually relaxing criterion:

$$pixel = \begin{cases} fore; \alpha \geq 0 * (1-k^i) + (0.5 + th\_NN) * k^i \\ \quad \& \ \alpha \geq 0 * (1-k^i) + \alpha 0 * k^i \\ back; \alpha \leq 1 * (1-k^i) + (0.5 - th\_NN) * k^i \\ \quad \& \ \alpha \leq 1 * (1-k^i) + \alpha 0 * k^i \end{cases} \quad (5)$$

in which k is the relaxing parameter. Lower k means the criterion is more relaxed and results in faster speed but is prone to assign foreground/background incorrectly, and thus k between 0.5 and 0.9 is recommended, i starts from 1 and is increased by one each iteration, until every pixel in the image is assigned foreground/background.

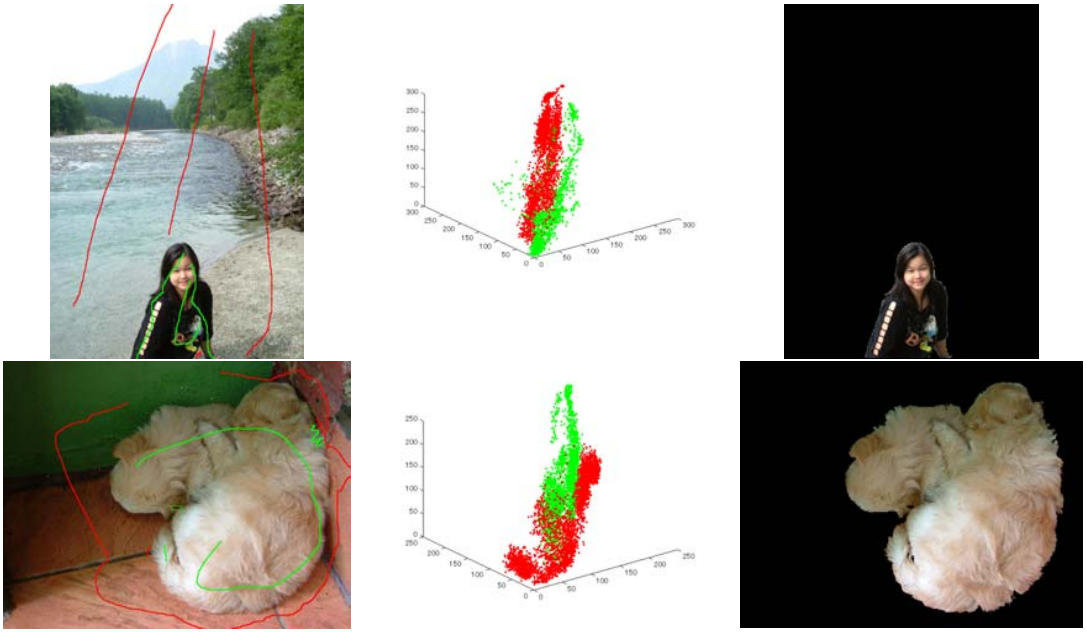
B.Robustness

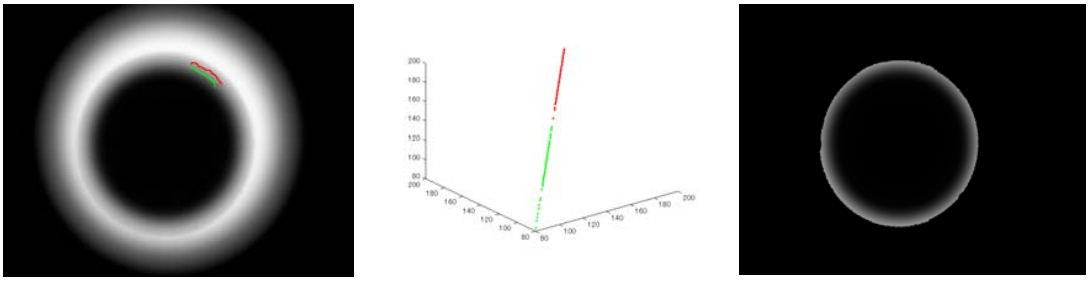
Unlike using statistical models to calculate the likelihood to be foreground/background, such as the Gaussian Mixture Model used in [15] or the K-means model used in [16], this neural network does not pre-assume the distribution function of the data. This is one reason that a neural network is superior to such methods in the case that the distribution of data is unknown. Such is the case in the second column of Fig.3; its color distribution would be poorly modeled by a presumed distribution such as Gaussian. Another reason for the method's robustness is that not only color is exploited as training data, but also texture.

If foreground/background characteristics are quite separated and gradient information is clear, the segmentation can be done without difficulty, as shown for example in row 1 of Fig. 3. Even the case that foreground/background characteristics are similar and gradient information is not clear, the boundary is well separated as shown in row 2 of Fig. 3.

For the case in which the foreground/background characteristics are actually the same, such as in row 3 of Fig. 3, if they do not touch each other, the neural network will not search in the area of the foreground that has the characteristics of the background, and vice versa. In this situation, it stops learning and the relaxing criterion, (5), would solve the rest of problem.

Even when the two areas around a boundary are very similar, the foreground/background are separated well as shown in Fig. 4 (c), compared to the graph cut implementation of [16] shown in Fig. 4 (b).





(a) (b) (c)

Fig. 3: Result of hybrid segmentation method. (a) User marked data. (b) The color distribution of marked pixels (the axis is red, green, and blue, respectively). (c) The recognized foreground.

(a) (b) (c)

Fig. 4: Comparison of segmentation tools. (a) User marked data. (b) Graph cut implementation of [16]. (c) Hybrid segmentation method.

For a 640x480 image, the overall process typically takes less than 10 seconds on a laptop with 2GHz CPU.

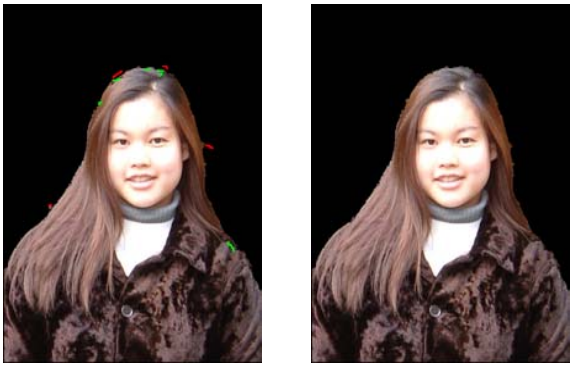
C.Boundary Editing

In the case that the foreground/background characteristics are very similar and the gradient information around the boundary is not clear, as in Fig. 4, existing methods could fail to adequately perform segmentation. To correct the boundary, [16] represents the border as a set of vertices that the user can edit. However, this method cannot deal with thin structures.

In this work, a simpler editing method that does not depend on the object shape is used. As the color segmentation is used to segment the original image into small regions, and used along with (4) to do foreground segmentation, the color-segmented image is exploited again here.

If the segmented foreground contains error, the user marks the output image as shown in Fig. 5 (a). The algorithm then performs region growing based on color-segmented image data, results are shown in Fig. 5 (b).

Though the method is very fast and simple, it relies heavily on the pre-segmentation image. The editing method that is fast as well as reliable is yet to be found.



(a) (b)

Fig. 5: Boundary editing. (a) User-mark data. (b) Region growing result.

III. Foreground Segmentation for Multi-Sensor based Object Modeling

In many areas, such as computer graphics, architecture and robotics, 3D modeling is currently used to generate virtual objects. To build a 3D model, a target object is scanned from multiple views by a laser range sensor, such as the Vivid 910 used in this work. The laser range sensor can output range and color images from the identical view simultaneously. Multiple range images are aligned and merged to compose geometric aspect of the 3D model. Then, color images are used to add textures of the 3D model.

Generally, each view's image consists of

foreground/background. The user must delete the background manually, usually using the range image, before exploiting the foreground images to generate the 3D model.

Instead of having to delete the background manually, foreground segmentation can be used directly to assist the current technology of multi-sensor modeling by segmenting the color image of each view and applying the result to the corresponding range image, as shown in Fig. 6. It is not always appropriate to use range information as an input to the neural network since it could result in bad training data, as shown in Fig.7, which would confuse the classifier.

The example of 3D model generated from segmented color and range images is shown in Fig. 8.

(a) (b)

Fig. 6: Foreground segmentation of each view. (a) User-marked data. (b) Hybrid segmentation result.

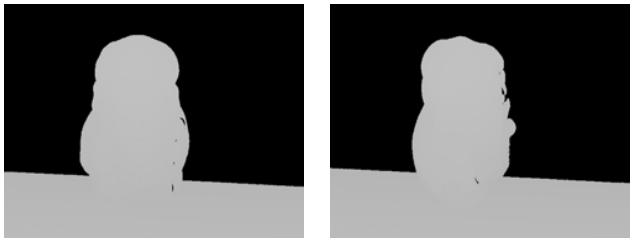


Fig. 7: Range data images from multiple views.

Fig. 8: Example of 3D modeling.

IV. Conclusion

This paper focuses on the problems of foreground segmentation and its applications in multi-sensor based object modeling.

For foreground segmentation, a hybrid system which exploits machine learning and techniques from graph cut works is proposed. The powerful machine learning technique of a neural network is used to recognize an input image's foreground/background color and texture characteristics. Neural network does not pre-assume the data distribution, and thus is flexible and robust. Instead of considering the quality of boundary data as a parameter in an optimization as in graph cut, it is considered as a constraint to temporarily stop the growing process. The main shortcoming is that it cannot deal with long, thin structures, e.g. hair, very well.

In 3D modeling, using foreground segmentation to extract the object of interest is also proposed to replace the tedious task of manually deleting the background. However, at present, a user must mark the foreground/background. One future research direction is to create an algorithm that requires less user interaction.

Reference

[1] Y. Y. Chuang, B. Curless, D. Salesin, and R. Szeliski, "A Bayesian approach to digital matting," *Proc. CVPR*, 2001.

[2] E. Mortensen and W. Barrett, "Intelligent scissors for image composition," *Proc. SIGGRAPH*, 1995.

[3] Y. Boykov and M. P. Jolly, "Interactive graph cuts for optimal boundary and region segmentation of objects in N-D images," *Proc. ICCV*, 2001.

[4] D. Greig, B. Porteous, and A. Seheult, "Exact MAP estimation for binary images," *J. Roy. Stat. Soc. B. 51*, 1989, 271–279.

[5] J. Wang and M. F. Cohen, "An iterative method approach for unified image segmentation and matting," *Proc. ICCV*, 2005.

[6] X. Feng, C. K. I. Williams, and S. N. Felderhof, "Combining belief networks and neural network for scene segmentation," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 24, 2002.

[7] J. Win and N. Jojic, "LOCUS: learning object classes with unsupervised segmentation," *Proc. ICCV*, 2005.

[8] D. R. Martin, C. C. Fowlkes, and J. Malik, "Learning to detect natural image boundaries using local brightness, color, and texture cues," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 26, 2004.

[9] J. A. Fails and D. R. Olsen, "A design tool for camera-based interaction," *Proc. CHI*, 2003.

[10] Y. Boykov and V. Kolmogorov, "An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 26, 2004.

[11] H. Luo and A. Eleftheriadis, "Model-based segmentation and tracking of head-and-shoulder video objects for real time multimedia services," *IEEE Trans. on Multimedia*, vol. 5, 2003.

[12] J. Wang, P. Bhat, R. A. Colburn, M. Agrawala, and M. F. Cohen, "Interactive video cutout," *Proc. SIGGRAPH*, 2005.

[13] Y. Li, J. Sun, and H. Y. Shum, "Video object cut and paste," *Proc. SIGGRAPH*, 2005.

[14] M. McGuire, W. Matusik, H. Pfister, J. F. Hughes, and F. Durand, "Defocus video matting," *Proc. SIGGRAPH*, 2005.

[15] C. Rother, V. Kolmogorov, and A. Blake, "GrabCut–interactive foreground extraction using iterated graph cut," *Proc. SIGGRAPH*, 2004.

[16] Y. Li, J. Sun, C. K. Tang, and H. Y. Shum, "Lazy snapping," *Proc. SIGGRAPH*, 2004.