Contour Detection

Project Midterm Report

Haiyan Yang, Shuo Han, Yuting Liu

1 Introduction

Contour detection of objects is essential in human vision. With damages in certain brain area responsible for the perception of contours, a person will be unable to recognize objects [1]. To develop a more powerful computer vision system, contour detection is also a must. In fact, contour detection is one of the most fundamental problems in computer vision. People can have a deep knowledge of the structural information in the images based on object contours, which many other tasks such as image segmentation, object recognition and scene understanding rely heavily on. Due to the importance of contour detection, among these years a lot of related work has been done and many methods have been developed to achieve better results.

In this project, our goal is to investigate how to detect contours. Our future work mainly focuses on two parts: (1) One is to study the existing methods comprehensively, implement them and compare their performances. (2) The other is to find a niche to improve current algorithms, or to apply the state-of-the-art techniques to other possible areas in computer vision.

2 Related Work

To evaluate contour detection algorithms in a quantitative manner, we will be using a dataset and its corresponding benchmark called Berkeley Segmentation Data Set and Benchmarks 500 (BSDS 500) [2]. It consists of natural images and their corresponding human-annotated ground truth contours. The benchmark includes the evaluation results of almost all modern approaches to contour detection.

A number of recent methods with satisfactory performance are based on the posterior probability of boundary (Pb) developed Martin et al. [3], which uses brightness, color and textures to detect both global and local contours. These techniques include the conditional random fields model developed by Ren et al. [4], the untangling cycle algorithm designed by Zhu et al. [5] and the idea of globalized

probability boundary (gPb) raised by Maire et al. [6]. Other progressive work not based on Pb includes the min-cover approach developed by Felzenszwalb and McAllester [7], the boosted edge learning algorithm invented by Dollar et al. [8], the method using sparse code gradients (SCG) proposed by Ren and Bo [9] and sketch tokens designed by Lim et al. [10].

3 Technical Thoughts

3.1 Probability of Boundary (Pb)

Early approaches of detecting contours aimed at finding sharp discontinuities in the brightness channel. But brightness alone is not enough. Recent methods of contour detection also included color and texture information to locate the correct boundaries. How to represent the changes in brightness, color and texture and how to combine them to predict the probability of boundaries are the problems which Pb focused on. Gradient operators for brightness, color and texture are introduced and some learning techniques are used to combine the gradients.

As for gradient, it is computed in the following way: draw a circle around a point and divide it two at some orientation, then the gradient is the chi-square difference between the two halves. This can be obtained by convolving the image with a half disk mask. Brightness and color are familiar concepts for us, but we still need to know how to represent texture. Textures are some patterns represented by clustering the responses with some filters. The responses are achieved by convolving the image with a collection of orientated derivative Gaussian filters and k-means algorithm is used to cluster the pixels.

After computing the gradients for brightness, color and texture, we can treat the combination problem as a supervised learning problem since the ground-truth contours were provided in BSDS 500. The following classifiers can be used: density estimation, classification trees, logistic regression, hierarchical mixtures of experts and support vector machines. Moreover, the local cues can be combined with global cues called gPb method.

3.2 Sparse Code Gradient (SCG)

Sparse coding is to learn a basis of a given training set to enable pixel representation and aggregation of pixel information in local neighborhoods. Efficient sparse coding and dictionary learning algorithms are performed with Orthogonal Matching Pursuit (OMP) and K-SVD to improve the gPb algorithm. The learning process will be separately applied to different channels on the training set including grayscale, chromaticity and surface normal.

K-SVD generalizes k-means and learns dictionaries of codewords from unsupervised data. Given a set of image patches $Y = [y_1, ..., y_n]$, K-SVD jointly finds a dictionary $D = [d_1, ..., d_m]$ and an associated sparse code matrix $X = [x_1, ..., x_n]$ by minimizing the reconstruction error:

$$\min_{a,b} \|Y - DX\|_F^2 \quad \text{s.t. } \forall i, \|x_i\|_0 \leq K; \forall j, \|d_j\|_2 = 1$$

In our implementation, the dictionary learning is established with patch-based pixel representation.

We choose patch size of 5×5 to represent every pixel in the image with dictionary size of 75.

With results of aggregate pixel information, our contour detection is based on oriented half-discs used in gPb. Two half-discs N^a and N^b of size $s \times (2s+1)$ for each of 8 orientations at each pixel p and scale s are defined for the contour classification. The final representation of contrast at a pixel p is the concatenation of sparse codes pooled at all scales $s \in \{1, ..., S\}$ ($S = 2$ or $S = 4$ is commonly used):

$$D_p = [D(N_1^a, N_1^b), ..., D(N_S^a, N_S^b); F(N_1^a \cup N_1^b), ..., F(N_S^a \cup N_S^b)]$$

Here the union term $F(N_1^a \cup N_1^b)$ denotes the whole appearance of the whole disk (including two half discs) and is normalized by $\|F(N_s^a)\| + \|F(N_s^b)\| + \varepsilon$.

We set a threshold α to decide whether or not the given pixel with its corresponding value will be classified as the edge.

3.3 Sketch Tokens

Sketch tokens is also an interesting method which achieves similar performance but uses much shorter time compared to other modern methods. They are supervisedly learned using mid-level information in the form of hand drawn contours in images. Patches of human generated contours are clustered to form sketch token classes and a random forest classifier is used for efficient detection in test images.

Technically speaking, sketch tokens are defined by classes of clustering patches extracted from binary images. Patches with a fixed size of 35×35 are extracted only if their center pixels indicate a contour. Clustering is done on Daisy descriptors computed over patches using k-means algorithm with $k = 150$ in the paper.

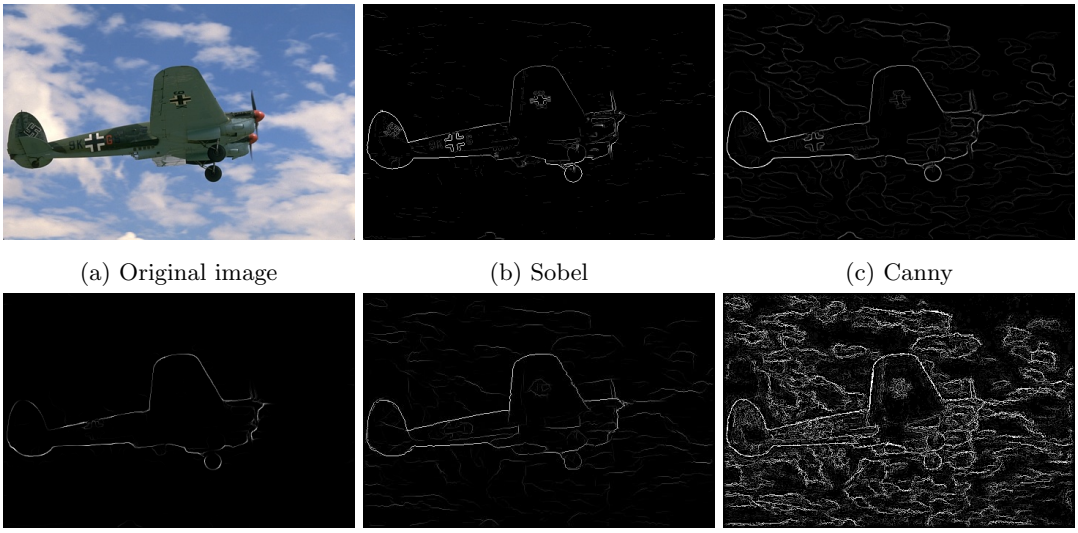
Given a set of sketch token classes, a random forest classifier is trained to detect their occurrence in color images. Features of the classifier are computed from patches into two types: channels and self-similarities.

Channels are composed of color, gradient and oriented gradient information in a patch. Three color channels are computed using the CIE-LUV color space. Three gradient magnitude channels are computed with different Gaussian blurs ($\sigma = 0, 1.5, 5$) and for the first two the gradient magnitude channels are split to create four additional channels each for a total of eight oriented magnitude channels. All channels are Gaussian blurred with $\sigma = 1$ and all the values in the resulting channels compose the first type of the feature.

Self-similarities are computed on a $m \times m$ grid ($m = 5$ in the paper) over the patch in each channel in the first type, thus for a 35×35 patch it yields 7×7 cells. For channel k and grid cells i and j, the self-similarity feature f_{ijk} is defined as $f_{ijk} = s_{jk} - s_{ik}$ where s_{jk} is the sum of grid cell j in channel k. The total number of such features is given by $\binom{m \cdot m}{2}$ due to symmetry.

Labels of the classifier are supplied by the clustering results if the patch is centered on a contour in the hand drawn sketches, otherwise the patch is assigned with a no-contour class.

After training the classifier, the probability of a contour at the center pixel of a patch in the test image



(a) Original image (b) Sobel (c) Canny

(d) Pb (e) SCG (f) Sketch tokens

Figure 1: Comparison of different contour detection methods applied on an example image.

is predicted if the patch belongs to any classes of sketch tokens, represented by $1 - P(\text{no-contour})$. Once the probability of a contour has been computed, a standard non-maximal suppression scheme is applied to find the peak response of a contour.

4 Preliminary Results

For Pb, a simplified version implementing the algorithm is completed. Our version includes a filter generator to compute the texture, a series of oriented half disc masks to compute the gradient and a function to compute the texture using k-means clustering. Based on our implementation, Pb is estimated as the product of the result of Canny method, gradients of brightness, color and texture. The Canny results serves as a non-maximum suppression scheme.

For SCG, the learning process combining K-SVD and OMP has finished with benchmark of BSDS 500 based on the gPb algorithm. Multi-scale neighborhoods coding is implemented for detection of contours with scale $S = 4$. The combination results of contour detection will be compared between gPb and SCG to see how performances are improved with dictionary learning.

For sketch tokens, due to the limitation of our computing power, in our implementation we reduce patch size to 21×21 and number of clusters to $k = 20$. Furthermore we used less training examples ($\sim$80k) to train the random forest classifier. Our implementation yields an F-score of 0.70, compared to 0.73 in the original paper.

Comparison of the contour detection results among various methods is shown in Figure 1. Note that comparing to traditional methods (Sobel and Canny), recent methods capture more contour information in human sense.

5 Remaining Milestones

5.1 Goals

For Pb:

– Use integral images to speed up the function to compute gradients.

– Try different filters to compute the texture.

– Try different distances instead of chi-square distance.

– Use some learning methods to combine different cues.

For SCG:

– Try different training set sizes and dictionary sizes to compute the accuracy of contour detection.

– Try different benchmarks to see whether performance will be influenced by images.

– Compare with our gPb implementation to see how performance is improved or not with dictionary learning process.

For sketch tokens:

– Try to find out more useful features other than channels and self-similarities.

– Try to further reduce the computational complexity of this method.

– Apply this method to broader areas, such as image segmentation and medical image analysis.

5.2 Important Dates

Date	Progress
Apr 1 – Apr 18	Original work (improvement and other applications)
Apr 19 – Apr 24	Presentation preparation
Apr 25 – May 5	Presentation and final report writing
May 6	Final report due

References

[1] Giuseppe Papari and Nicolai Petkov. "Edge and line oriented contour detection: State of the art". In: *Image and Vision Computing* 29.2 (2011), pp. 79–103.

[2] Pablo Arbelaez, Michael Maire, Charless Fowlkes, et al. "Contour detection and hierarchical image segmentation". In: *Pattern Analysis and Machine Intelligence, IEEE Transactions on* 33.5 (2011), pp. 898–916.

[3] David R Martin, Charless C Fowlkes, and Jitendra Malik. "Learning to detect natural image boundaries using local brightness, color, and texture cues". In: *Pattern Analysis and Machine Intelligence, IEEE Transactions on* 26.5 (2004), pp. 530–549.

[4] Xiaofeng Ren, Charless C Fowlkes, and Jitendra Malik. "Scale-invariant contour completion using conditional random fields". In: *Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on*. Vol. 2. IEEE. 2005, pp. 1214–1221.

[5] Qihui Zhu, Gang Song, and Jianbo Shi. "Untangling cycles for contour grouping". In: *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on*. IEEE. 2007, pp. 1–8.

[6] Michael Maire, Pablo Arbeláez, Charless Fowlkes, et al. "Using contours to detect and localize junctions in natural images". In: *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*. IEEE. 2008, pp. 1–8.

[7] Pedro Felzenszwalb and David McAllester. "A min-cover approach for finding salient curves". In: *Computer Vision and Pattern Recognition Workshop, 2006. CVPRW'06. Conference on*. IEEE. 2006, pp. 185–185.

[8] Piotr Dollar, Zhuowen Tu, and Serge Belongie. "Supervised learning of edges and object boundaries". In: *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on*. Vol. 2. IEEE. 2006, pp. 1964–1971.

[9] Xiaofeng Ren and Liefeng Bo. "Discriminatively trained sparse code gradients for contour detection". In: *Advances in neural information processing systems*. 2012, pp. 584–592.

[10] Joseph Lim, C Zitnick, and Piotr Dollár. "Sketch tokens: A learned mid-level representation for contour and object detection". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2013, pp. 3158–3165.