Comparison of Edge Detectors

Jong-Chyi Su
University of California, San Diego
9500 Gilman Dr., La Jolla, CA
jcsu@eng.ucsd.edu

Abstract

Edge detection is a fundamental task in computer vision. In this assignment, I compared six different algorithms for edge detection, including Sobel, Canny [2], BEL [3], gPb [1], Sketch Token [5], and Structured Forest [4]. The Berkeley Segmentation Dataset (BSDS300) [6] and benchmark are used to compare the edge detectors. For image processing methods, Canny and Sobel, the precisions are below 50% *though their frame rate can up to 20 and 200 FPS. For learning based algorithms, gPb, Sketch Token, and Structured Forest have the precision above 70%. while Structured Forest has the best performance on speed.*

1. Introduction

Edge detection is a fundamental task in computer vision. Although it has been studied for decades, it is still very challenging. The Sobel and Canny edge detector, which are invented in 1980s, still have been widely used. Recently, many learning based edge detection methods have been proposed. Instead of using only low-level cues like Sobel and Canny detectors, combining low, middle, and high level information by machine learning methods are proposed. In BEL [3], a probabilistic boosting tree is used. In Sketch Token [5] and Structured Forest [4], random forest have been used for supervised learning. In addition, the gPb [1] method combines contour detection and spectral clustering method.

In this assignment, those six edge detectors are compared. However, the edge detection itself is an ill-posed problem. To give a benchmark of edge detectors, the Berkeley Segmentation Dataset (BSDS300) [6] provides 200 training images and 100 test images with human annotated ground truth boundaries. To measure the benchmark, instead of traditional ROC curve, they use precision and recall curve. Both precision and recall curves and speed are measured in the experiment. The Sketech Token has the best performance while the Structured Forest is the fastest learning based edge detector.

2. Edge Detectors

2.1. Sobel and Canny

The Sobel and Canny edge detectors are both based on gradient image. The first step is the convolution of image and a gradient kernel on x and y direction. After thresholding the image, it is the result of Sobel detector. For Canny detector, non-maximum suppression and hysteresis thresholding are also used for tracing along the edge.

2.2. BEL

The concept of Boosted Edge Learning (BEL) is to combine low, middle, and high level cues of the image. Like the concept of AdaBoost, weak classifiers can be combined in a cascade way. The decision tree with boosting, which is called probabilistic boosting tree, is used. Manually labeled positive and negative image patches of edges are used for training. For every tree node, classifiers with approximately 50000 features are trained by sampling those patches.

2.3. gPb

The globalPb (gPb) contour detector combines bottom-up gradient based edge detector and a global spectral clustering method. First, a multiscale posterior probability function is used by measuring gradients on brightness, color, and texture. After finding bottom-up edges as local signals, use spectral clustering to further refine the edges. The affinity map is build by the posterior probability. Find the corresponding eigenvecors of top eigenvalues, then treat the gradient of those eigenvectors as spectral signals. The final results is a weighted sum of local and spectral signal.

2.4. Sketch Token

The Sketch Token can be regards as a mid-level features learned from manually annotated contour images. The edge patches from hand drawn sketches are clustered to form sketch tokens. Random forest classifier are then used to train the edge detector. A random forest is a voting for the result of different decision trees. Each tree are trained by

contour positive patches, which belongs to different token classes, and "'no contour"' negative patches.

2.5. Structured Forest

In Structured Forest, the structured learning framework with random decision forests is used. Like Sketch Token, the patches of edges are used to train random forests. To be more effective, they map structured labels to a discrete set of labels prior to training. In addition, a intermediate mapping is used for the convenience of calculating Euclidean distances to approximate the dissimilarity of structured labels.

3. Experiment

To evaluate the performance of edge detectors, I used the Berkeley Segmentation Dataset (BSDS300) and also its benchmark method described in [6]. The benchmark is measured in precision and recall curve. The ground truth boundaries are given by human segmented images. The edge detectors will generate a soft boundary map, where each pixel value represents the probability of being edges. Then, the precision and recall value can be calculated. The precision is a measure of false positives while the recall is a measure of the percentage of true positives. In addition to the precision-recall curve, the F-measure score, which is the harmonic mean of the curve, is also been calculated.

4. Result

For the timing issue, I only choose ten pictures from BSDS 300 test image sets in the experiment. The precision-recall curve in Figure 1 shows that gPb, Sketch Token and Structured Forest have about the same performance. Note that for Canny and Sobel, I used one set of fixed parameters for all images. The result is a binary map but a soft map so there is only one point in the precision-recall curve. Table 1 shows the F-measure value and the speed (in FPS) of six algorithms. Note that I did not fine-tune the parameters of those algorithms. The Sketch Token has the best performance in F-measure score. Non-learning based algorithms like Canny and Sobel are fast but not accurate. Among learning based algorithms, Structured Forest is the fastest one. The experiment is run on the laptop with Intel i-5 core. Some results images of edge detectors are shown in Figure 2 3 4 5.

5. Conclusion

Edge detectors has been an important and fundamental issue of computer vision for decades. In the beggining, the low-level gradient cues are used for Sobel and Canny edge detector. In gPb, spectral clustering results are combined, but with high computational complexity. Recently, supervised learning method are used in edge detection like BEL, Sketch Token and Structured Forest. The later can achieve real-time while retaining good quality.

Method	F-measure	FPS
Human	.79	-
Canny	.49	20
Sobel	.40	100
BEL	.63	$1/50$
gPb	.71	$1/720$
SketchToken	.73	1.5
StructuredForest	.71	8

Table 1. The F-measure score and the speed comparison of edge detectors.

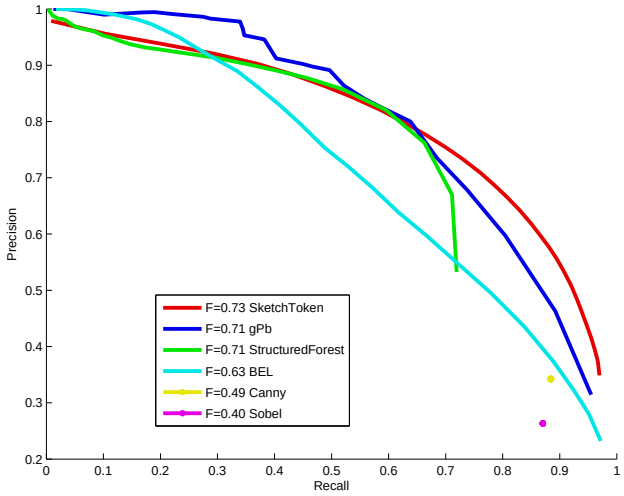


Figure 1. The precision and recall curve of edge detectors.

References

[1] P. Arbelaez, M. Maire, C. Fowlkes, and J. Malik. Contour detection and hierarchical image segmentation. *PAMI*, 33, 2011.

[2] J. Canny. A computational approach to edge detection. *PAMI*, 8(6):679698, November 1986.

[3] P. Dollr, Z. Tu, and S. Belongie. Supervised learning of edges and object boundaries. *CVPR*, 2006.

[4] P. Dollr and C. Zitnick. Structured Forests for Fast Edge Detection. *ICCV* 2013

[5] J. Lim, C. L. Zitnick, and P. Dollar. Sketch tokens: A learned mid-level representation for contour and object detection. *CVPR*, 2013.

[6] A Database of Human Segmented Natural Images and its Application to Evaluating Segmentation Algorithms and Measuring Ecological Statistics. *Proc. ICCV*, 2, 2001.

(a) Original Image (b) Human

(c) Sobel (d) Canny

(e) BEL (f) gPb

(g) Sketch Token (h) Structured Forest

Figure 2. Result image set 1.

(a) Original Image (b) Human

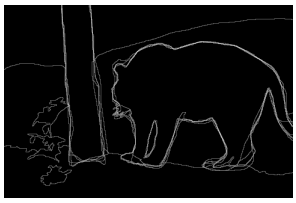
(c) Sobel (d) Canny

(e) BEL (f) gPb

(g) Sketch Token (h) Structured Forest

Figure 3. Result image set 2.

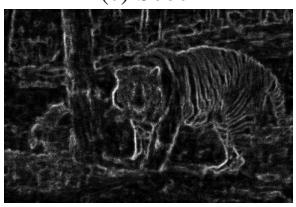
(a) Original Image



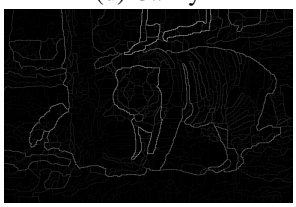
(b) Human

(c) Sobel

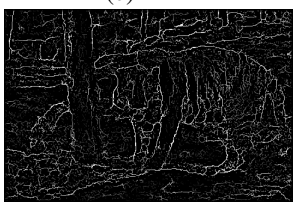
(d) Canny



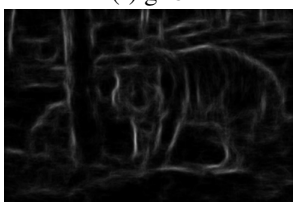
(e) BEL



(f) gPb



(g) Sketch Token



(h) Structured Forest

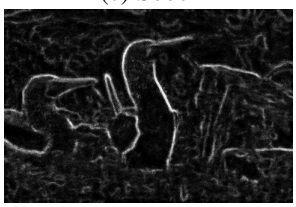
Figure 4. Result image set 3.

(a) Original Image

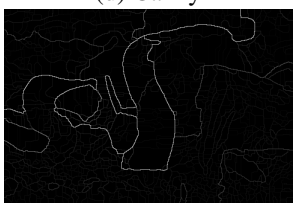
(b) Human

(c) Sobel

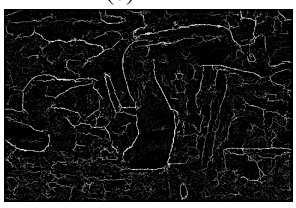
(d) Canny



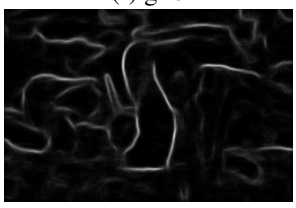
(e) BEL



(f) gPb



(g) Sketch Token



(h) Structured Forest

Figure 5. Result image set 4.