Sketch Tokens: A Learned Mid-level Representation for Contour and Object Detection

Joseph J. Lim  
Massachusetts Inst. of Technology  
lim@csail.mit.edu

C. Lawrence Zitnick  
Microsoft Research  
larryz@microsoft.com

Piotr Dollár  
Microsoft Research  
pdollar@microsoft.com

Abstract

We propose a novel approach to both learning and detecting local contour-based representations for mid-level features. Our features, called sketch tokens, are learned using supervised 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 novel images. We demonstrate our approach on both top-down and bottom-up tasks. We show state-of-the-art results on the top-down task of contour detection while being over 200× faster than competing methods. We also achieve large improvements in detection accuracy for the bottom-up tasks of pedestrian and object detection as measured on INRIA [2] and PASCAL [4], respectively. These gains are due to the complementary information provided by sketch tokens to low-level features such as gradient histograms.

1. Introduction

For visual recognition, mid-level features provide a bridge between low-level pixel-based information and high-level concepts, such as object and scene level information. Effective mid-level representations abstract low-level pixel information useful for later classification while being robust to irrelevant and noisy signals.

In this work, we propose a novel approach to both learning and detecting local edge-based mid-level features, and demonstrate their effectiveness for both bottom-up and top-down tasks. Our features, called sketch tokens, capture local edge structure. The classes of sketch tokens range from standard shapes such as straight lines and junctions to richer structures such as curves and sets of parallel lines (Fig. 1).

Given the vast number of potential local edge structures, we must select an informative subset to represent by the sketch tokens. We propose a method for discovering these classes using human-labeled sketches [1].

2. Sketch Tokens

2.1. Defining sketch token classes

Our goal is to define a set of token classes that represent the wide variety of local edge structures in an image. These include straight lines, t-junctions, y-junctions, corners, curves, parallel lines, etc. We propose a method for discovering these classes using human-labeled sketches [1].

Let us assume we have a set of images with a corresponding set of binary images \( S \) representing the hand drawn contours. We define the set of sketch token classes by clustering patches \( s \) extracted from \( S \). Example cluster means are illustrated in Figure 1. Notice the variety of the sketch tokens, ranging from straight lines to more complex structures.

2.2. Detecting sketch tokens

Given a set of sketch token classes, we wish to detect their occurrence in color images. As input, features are computed from color patches \( x \) extracted from the training images \( I \). Ground truth class labels are supplied by the clustering results described above.

Feature extraction: For feature extraction, we use an approach inspired by Dollár et al. [3] and compute multiple feature channels per image. Two types of features are then employed: features directly indexing into the channels and self-similarity features.

Our channels are composed of color, gradient, and oriented gradient information in a patch extracted from a color
image. Three color channels are computed using the CIELUV color space. We compute several normalized gradient channels that vary in orientation and scale [3, 2].

The second type of feature used by our method is based on self-similarity. The self-similarity features capture the portions of an image patch that contain similar textures based on color or gradient information. More details on how we capture the self-similarity can be found in [6].

**Classification:** Two considerations must be taken into account when choosing a classifier for labeling sketch tokens in image patches. First, every pixel in the image must be labeled, so the classifier must be efficient. Second, the number of potential classes for each patch ranges in the hundreds. In this work we use a random forest classifier, since it is an efficient method for multi-class problems.

### 3. Experimental Results

**Contour detection results:** Sketch tokens provide an estimate of the local edge structure in a patch. The details on how to compute the binary labeling of pixel contours from mid-level sketch tokens can be found in [6].

We test our contour detector on the BSDS500 [1]. In Table 1, we compare our method against competing methods. Our method achieves state-of-the-art results among all local methods, and achieves nearly the accuracy of global methods. Our method achieves state-of-the-art results among all local methods, and achieves nearly the accuracy of global methods. Our method achieves state-of-the-art results among all local methods, and achieves nearly the accuracy of global methods.

![Sketch tokens](image)

**Table 1.** Contour detection result on BSDS500: We achieve state-of-the-art results among all local methods. The methods shown in the last two rows perform complex global resulting in slightly better performance. However, our approach is 240-280x faster.

<table>
<thead>
<tr>
<th>Method</th>
<th>ODS</th>
<th>OIS</th>
<th>AP</th>
<th>Speed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Human</td>
<td>0.80</td>
<td>0.80</td>
<td>1.0</td>
<td>-</td>
</tr>
<tr>
<td>Canny</td>
<td>0.60</td>
<td>0.64</td>
<td>0.58</td>
<td>1/15 s</td>
</tr>
<tr>
<td>gPb (local)</td>
<td>0.71</td>
<td>0.74</td>
<td>0.65</td>
<td>60 s</td>
</tr>
<tr>
<td>SCG (local)</td>
<td>0.72</td>
<td>0.74</td>
<td>0.75</td>
<td>100 s</td>
</tr>
<tr>
<td>Sketch tokens</td>
<td>0.73</td>
<td>0.75</td>
<td>0.78</td>
<td>1 s</td>
</tr>
<tr>
<td>gPb (global)</td>
<td>0.73</td>
<td>0.76</td>
<td>0.73</td>
<td>240 s</td>
</tr>
<tr>
<td>SCG (global)</td>
<td>0.74</td>
<td>0.76</td>
<td>0.77</td>
<td>280 s</td>
</tr>
</tbody>
</table>

Table 2. Accuracy of [3] combined with sketch tokens on INRIA pedestrian with varying choice of channels.

<table>
<thead>
<tr>
<th></th>
<th>channels</th>
<th># channels</th>
<th>miss rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>HOG</td>
<td>27.9</td>
<td>10</td>
<td>58.2</td>
</tr>
<tr>
<td>ST+HOG</td>
<td>23.8</td>
<td>151</td>
<td>19.5%</td>
</tr>
</tbody>
</table>

Our approach using 150 sketch tokens achieves a MR of 19.5%. Combining sketch tokens and the 10 low-level features achieves 14.7%.

**PASCAL VOC 2007:** Our final set of results use the PASCAL VOC 2007 dataset [4]. We perform experiments with the deformable parts model (DPM) of Felzenszwalb et al. [5]. We propose adding our sketch tokens to the HOG features for training the DPMs. Results are shown in Table 3 for DPMs. In nearly all cases top Average Precision (AP) scores are achieved with a combination of HOG and sketch tokens. This demonstrates that sketch tokens may provide a complementary information over the HOG descriptor.

### References


