HOG Features on the GPU

1 Introduction

Histogram of Oriented Gradients (HOG) features are a trending topic in object detection literature. HOG features are a robust way of describing local object appearances and shapes by their distribution of intensity gradients or edge directions, and have been used successfully as a low level feature in a number of object recognition tasks. Human faces are generally considered interesting and important to detect in many applications such as surveillance, recognition systems, biomedical, and video. HOG descriptors have been shown to significantly outperform existing feature sets for human detection, but at the expense of greater computational overhead than other well known Haar-like rectangular features. We propose to accelerate the HOG feature descriptor computation by exploiting GPU parallelism so that existing frameworks for real-time detection can make use of a more robust feature.

2 Related Work

The success of the Viola-Jones detector [3] illustrated the feasibility of real-time face detection. Their key to success was coupling a simple and fast-to-compute set of features with a machine learning algorithm that could perform the computationally complex task of learning offline. This sacrifice at the feature level can make the detector more sensitive to noise that a more robust — but computationally expensive – feature would avoid.

The HOG feature [1] has seen wide recognition as a successful feature for object detection [2], which we think can be successfully implemented on the GPU for a speed-up that allows for real-time detection comparable to the rectangular Haar-like detectors used by Viola-Jones. Results from Zhu et al. [4] suggest that HOG features are more accurate than the rectangular features found in Viola-Jones.

3 Implementation

The HOG feature is closely related to the SIFT feature descriptor, but while SIFT is intended to be run at a sparse set of interest points, HOG is intended to be run over a dense grid. We implemented the HOG feature as it was described in [1], where it was used in the context of pedestrian detection. First, the 2D gradient of the image is computed using a vertical and horizontal $[-1, 0, 1]$ filter. Then, the image is divided into M cells of$N \times N$ pixels. A histogram with H bins is computed and normalized given the

weighted gradient at each pixel, for each of the cells. The concatenation of the histograms from each cell yields a $H \times M$ length feature vector for the image. Figure 1 shows the gradient and histogram for a sample image. We implemented kernels for computing the gradient over the entire image, and for computing normalized histograms for each cell concurrently, and concatenating these into the feature vector used by the SVM.

Given a set of labeled training images, a window around the object to be detected can be extracted, and HOG computed over it. These feature vectors can then be used to train a SVM. We used the SVM-light package for training, and implemented our own SVM classifier as a kernel on the GPU. We extracted positive examples of faces using the labeled Caltech face data base[1], and selected random windows from the same dataset that did not overlap with the face regions for negative examples. This gave us a training set of more than 400 positive and 400 negative examples. To perform detection, we took a sliding window approach, computing the HOG feature for each window and passing it as input to our SVM classifier.

4 Evaluation

Our implementation of HOG runs in real time: 0.06 seconds to compute a HOG feature with 16×16 cells and 8 bins per histogram for a 640×480 frame. Unfortunately our naive sliding window approach to detection is much slower than we had hoped. OpenCVs CascadeClassifier can perform the detection task on average in 0.6 seconds, where as our approach takes roughly 6.0 seconds. On top of this significant performance hit, the SVM classifier was unable to successfully learn to discriminate between faces and non-faces.

[1] `http://www.vision.caltech.edu/html-files/archive.html`

5 Discussion

Our original premise was that we could efficiently compute HOG features using the GPU, and that using HOG features would improve classification performance. Unfortunately we were only able to show the former in this project. Our ability to compute HOG in real time is directly related to being able to decompose the image and work on individual cells simultaneously. The main reason for the slowdown comes from our naive sliding window approach which re-computes cells overlapped from previous windows. Caching these results would significantly improve our run time.

We believe the main reason we were unable to train the SVM successfully was due to our small training data set. Each feature vector is a concatenation of the histogram for each cell, so our feature vector had over 5000 components, which would require a much larger training set. In addition to that, one extension that was mentioned in [1] as improving performance was the use of block based normalization. Instead of normalizing each cells histogram individually, cells are grouped into blocks of four, and normalized over the entire block. Cells can be grouped into blocks of four in four different overlapping ways, so four different normaliza-

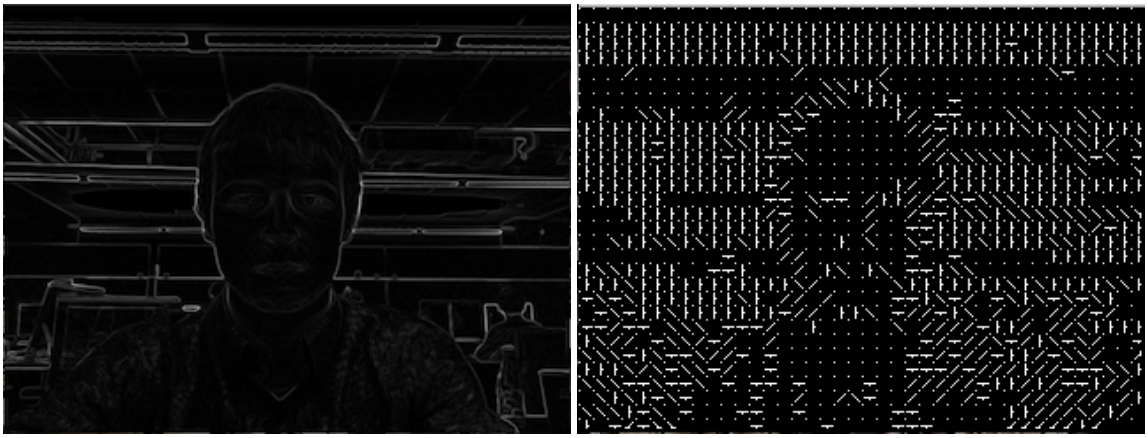

Figure 1: Gradient magnitude computed using 1D filters and maximum binned orientation per cell.

tions are computed for each cell, making the feature vector length four times longer. Our current dataset would be even less sufficient for this feature representation.

References

[1] N. Dalal, B. Triggs, and C. Schmid. Human detection using oriented histograms of flow and appearance. *Computer Vision–ECCV 2006*, pages 428–441, 2006.

[2] P. Felzenszwalb, D. McAllester, and D. Ramanan. A discriminatively trained, multiscale, deformable part model. In *Computer Vision and Pattern Recognition, IEEE Conference on*, pages 1–8. IEEE, 2008.

[3] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*, volume 1, pages I–511. IEEE, 2001.

[4] Q. Zhu, M. Yeh, K. Cheng, and S. Avidan. Fast human detection using a cascade of histograms of oriented gradients. In *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on*, volume 2, pages 1491–1498. IEEE, 2006.