SURF 2010 July Progress Report

Justin Johnson

July 1, 2010

1 Background

An important area of recent investigation in the field of computer vision has been that of visual recognition. When a human being looks at a still image or watches a video, it is easy for him or her to identify objects such as people, cars, and cell phones. The overall goal of visual recognition research is to develop algorithms and techniques which endow machines with the same ability. There has been considerable recent interest in the specific task of recognizing pedestrians, as it is believed that algorithms which can successfully recognize pedestrians can be generalized to other categories [5]. In addition, pedestrian detection itself has many potentially useful applications. For example, cars could be equipped with cameras and automatically stop when a pedestrian impedes their path, or surveillance cameras could automatically focus on pedestrians in a scene.

As discussed in [5], current methods of pedestrian detection yield promising results, but are still not accurate enough for real world applications. To date, the most successful pedestrian detectors run on single images [1]. To detect pedestrians in video, the detector is run on each frame and detections across frames are associated to form object trajectories [7]. The purpose of my project is to utilize temporal information in videos to increase the speed and accuracy of existing pedestrian detectors with the long term goal of implementing a system which can identify and track pedestrians in real time.

2 Previous Work

Prior work on pedestrian detection has resulted in code which can detect pedestrians in still images. In addition, prior to the start of the summer a dataset of video featuring nearby sidewalks was collected. This dataset contains 25 video clips, each approximately an hour in length. These videos will form the dataset on which our algorithms will be run.

3 Problem and Approach

As mentioned above, the purpose of the project is to use the additional information present in videos in order to better detect and track pedestrians. Currently the pedestrian detector works by by iterating over a dense set of subwindows of various sizes and shapes in each frame and deciding whether each subwindow contains a pedestrian. This process is slow; currently it takes about 5 seconds to detect the pedestrians in a single 1280x720 pixel frame. We have identified a few approaches for attacking this problem.

One general approach is to use additional information present in video to identify regions of each frame which have a high probability of containing pedestrians. The existing detector could then be modified to focus on subwindows only within these high-probability regions. For example, one could use existing methods to estimate the three dimensional geometry of the scene [3] and then only look for pedestrians in regions corresponding to roads or sidewalks.

Another approach is to make use of the temporal information present in videos. For example, any pedestrian in the video will most likely enter at some point in one frame, move for a number of frames, and then exit at a different point in a later frame. If the detector were to report an isolated detection which did not correspond to nearby detections in adjacent frames, then this detection could be a false positive. A similar approach could be used to identify missed detections.

4 Progress

A large portion of the last three weeks has been spent familiarizing myself with prior work and preexisting tools which relate to this project. I have become familiar with Matlab and have learned how to expedite the execution of numerically heavy code by distributing jobs over a number of computers. The videos collected prior to the beginning of the summer have been converted to a more convenient file format. The next step is to run the existing pedestrian detector on every frame of these videos. Once this is done a system for efficiently querying these precomputed detections needs to be written.

As a first step towards utilizing temporal information, work has begun on implementing the object tracking algorithm described in [7]. Given object detections in each frame of a video, this algorithm strings together detections to form object trajectories. This algorithm depends on an efficient method to solve minimum-cost network flow problems, so an existing implementation of a stan-

dard min-cost network flow algorithm was found [2] and code was written to allow this implementation to interface with Matlab.

5 Challenges and Resources

There have been no major challenges encountered so far. The main difficulties to this point have been related to the large amount of time needed to gain understanding of the work that has already been done and familiarity with the tools that have been developed. This trend will likely continue and be the main obstacle of the project.

At this time I do not require any resources to which I do not already have access, and do not anticipate the need for any additional resources.

References

[1] P. Dollár, Z. Tu, P. Perona, and S. Belongie, *Integral Channel Features*, BMVC, 2009.

[2] A.V. Goldberg, *An Efficient Implementation of a Scaling Minimum-Cost Flow Algorithm* 1*, Journal of Algorithms **22** (1997), no. 1, 1–29.

[3] D. Hoiem, A.A. Efros, and M. Hebert, *Putting objects in perspective*, International Journal of Computer Vision **80** (2008), no. 1, 3–15.

[4] S. Maji and J. Malik, *Object detection using a max-margin hough transform*, (2009).

[5] B. Schiele P. Dollar, C. Wojek and P. Perona, *Pedestrian detection: A benchmark*, CVPR, June 2009.

[6] P. Viola and M. Jones, *Rapid object detection using a boosted cascade of simple features*, IEEE Computer Society Conference on Computer Vision and Pattern Recognition, vol. 1, Citeseer, 2001.

[7] L. Zhang, Y. Li, and R. Nevatia, *Global data association for multi-object tracking using network flows.*