Fast Parallel Object Tracking

Timothy Li
April 10, 2017

1 Summary

We will implement a machine learning algorithm to track objects in video utilizing the parallel capabilities of graphics cards. In particular, we will use CUDA on the GHC machines to parallelize a boosting algorithm by Helmut Grabner, Michael Grabner, and Horst Bischof.

2 Background

The task of object tracking is the following: given a video and an identified object (usually given by a bounding box in the first frame), track the position of the object over the frames of the video. There are inherent factors that make object tracking easy. For instance, objects should not move too much from frame to frame which significantly decreases the search space. On the other hand, object tracking can be very hard: tracking must deal with various difficulties usually related to the video itself like lighting changes, object rotation, temporary or partial obstruction of the object, nonstationary camera, or changes in appearance of the object itself. Thus, it is important for object tracking algorithms to adapt and adjust to such changes.

The algorithm by Grabner, Grabner, and Bischof [2] accounts for this by using multiple “weak classifiers” and “selectors” that choose the strongest of the weak classifiers. The weak classifiers are chosen to be fast but not necessarily the most accurate which allows for performance. The accuracy is boosted and the algorithm adapts by combining the decisions of the selectors and constantly updating weights. The potential for parallelism appears in a couple places. First, the classifiers themselves can classify and update in parallel. Further, classification on a particular frame occurs many times over many possible bounding boxes which can also be done in parallel. This opens the door for very fast parallel execution.

3 The Challenge

There is an obvious dependency between frames i.e. the area of search for the object on the \( i \)th frame will depend on the location of the object on the \( i - 1 \)th
frame. However, this is not necessarily a strict dependency because we may choose to look at the \( i - k \)th frame instead and compute in batches of frames. This can decrease accuracy but may increase performance. As presented in the paper, there is also a dependency of selectors in training because they need to update their weights sequentially but this may not be a true bottleneck. Computation will likely be the dominating factor here because each frame of data requires significant amounts of computation to process. However, approaching realtime speeds may introduce a bandwidth bottleneck as frames are loaded in. Overcoming these difficulties and tuning the algorithm to be accurate and performant will be the main challenge in this project.

4 Resources

We will be using the GHC machines and their NVIDIA GeForce GTX 1080 graphics cards to do the heavy lifting. The code will be written from scratch. We will be using several papers for the algorithms. Grabner and Bischof [1] provide a version of the boosting algorithm. Grabner, Grabner, and Bischof [2] provide a version of the boosting algorithm using a “pool” of classifiers. In general, we will have to research and find algorithms for the classifiers themselves.

5 Goals and Deliverables

We plan on building an accurate and performant object tracker according to the specified algorithm built on top of a few weak classifiers. The object tracker should be able to track simple objects at reasonable speeds. According to [2], they were able to achieve 20 fps using 50 selectors and 250 weak classifiers at resolution \( 640 \times 480 \) on a standard 1.6 GHz PC. Thus, we can hope to use more selectors, more weak classifiers at a higher resolution and framerate simply because of the computing power available on the GHC machines. We hope that our implementation can reach realtime processing with comparable accuracy. Speed comparisons can be made against OpenCV which implements a similar tracking algorithm under the name BOOSTING. If we get ahead of schedule, we can implement more weak classifiers providing even more robust tracking.

It may be impossible to provide an interactive demo of the tracking software because of the logistics, but videos of the tracking should be shown. In addition, we should be able to provide graphs of the speedup in addition to comparisons to OpenCV.

6 Platform Choice

The NVIDIA GeForce GTX 1080 is a good choice for this problem because of its parallel capabilities. We will need to be able to leverage lots of computation in order to classify and train using the classifiers in realtime.
7 Schedule

- Week of 4/9: design allocation of workload and overall structure
- Week of 4/16: implement two weak classifiers
- Week of 4/23: implement selectors and the strong classifier
- Week of 4/30: finish implementation of the object tracker and begin running tests on GHC machines
- Week of 5/7: finish tuning of performance and accuracy, comparisons against OpenCV

References
