15-418 Project Checkpoint Report
Kunal Jobanputra (kjobanpu), Ryan Stentz (rastentz)

Project Checkpoint Report

Work Completed So Far

- As a general overview, we have implemented both the serial as well as the parallel version of the Harris Corner Detector. Upon some initial testing, they seem to be working correctly, although we will more intensively test our results in the coming weeks. The serial version seems to be outperforming the parallel version that we implemented in CUDA. As expected, the main bottleneck is the memory allocation as well as copying memory to and from the GPU. For specific numbers, follow this link: Midpoint Results. More specific week to week updates can be seen on the project website.

- Implementing the serial version allowed us to understand which parts of the algorithm are independent of each other. We identified that there were 3 distinct phases in the algorithm: convoling the Gaussian kernel, computing the gradients and getting the R score, and finally non-maximum suppression. We must complete each phase completely before moving on to the next, but each pixel computation can be done independently at each phase, making it well equipped as a GPU computation. However, as mentioned before, since there is a substantial bottleneck of the amount of memory that needs to be sent back and forth from the GPU, we will discuss future approaches in the next section.

Worked To Be Completed

- *Goals*
    - Improve the performance of the parallel algorithm:
        - Try to use all the data in one pipeline instead of storing intermediate results in between phases.
        - Eliminate need for padding and reason about data that is replicated when doing computation.
        - Compress output, if possible, so that data latency is reduced.
        - Evaluate other techniques to run the algorithm in a parallel fashion (e.g. shared address space through OpenMP)
    - Measure to see how our implementation translates to an embedded device, something commonly used in computer vision to see if our application could realistically be deployed.
- *Nice to have*
    - Implement homography estimation to take in matched image features and find transformations that relate them. This would employ our implementation of the Harris Corner Algorithm, which could be exciting.

Schedule

So far, we have kept on track with our schedule. It seems feasible that we will be able to meet all our goals. Our stretch goal may be a bit tougher to accomplish, depending on how long our more immediate goals take.

Week 5:

- (11/30 - 12/3):

    - Kunal: Look into using OpenMP for the parallel implementation
    - Ryan: Eliminate padding
- (12/4 - 12/6):
    - Ryan, Kunal: Eliminate storing intermediate results in between phases

Week 6:

- (12/7 - 12/10):
    - Kunal: Implement OpenMP version
    - Ryan: Test our how implementation translates to an embedded device
- (12/11 - 12/14):
    - Kunal, Ryan: Prepare for final presentation and demo
    - Kunal, Ryan: If time permits, work on homography estimation

Final Presentation

We plan to show images of how well the algorithm can find corners as well as sharing how fast serial and parallel versions of the algorithms run in. We will share performance results with graphs. We will briefly explain the algorithm on a high level as well as the parts of the algorithm that can run in parallel. We can talk about the challenges and how we overcame them. If we finish our stretch goal, we will speak about how the corner detector relates to it and the unique challenges associated with homography estimation.

Concerns

The following concerns are more of us just sitting down and doing the work. Nevertheless, concerns include the following:

- Figuring out if we can get a faster implementation using a GPU
    - Memory latency may prove to be too much of a bottleneck
- Getting the OpenMP version working
- Trying to finish to our stretch goal