Parallel Seam Carver

Abigail Li (abigail2), Jonathan Liu (jsliu2)

SUMMARY: Summarize your project in no more than 2-3 sentences. Describe what you plan to do and what parallel systems you will be working with.

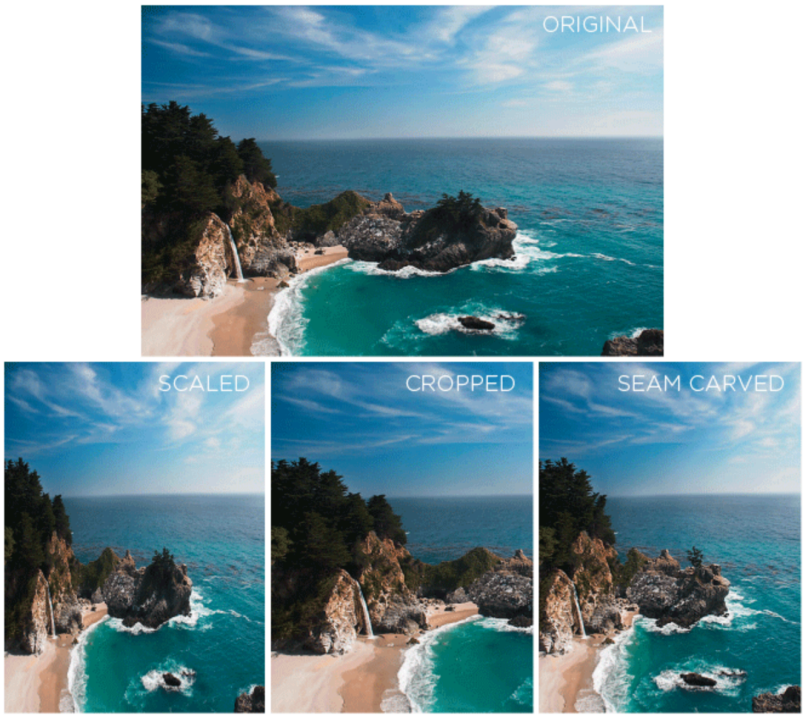
The Seam Carving algorithm is an algorithm that resizes an image by removing seams (paths from one end of the image to another) that have the least importance. One way of implementing this is via a Dynamic Program approach, so we will be trying to parallelize the DP version of the Seam Carving algorithm such that we can achieve at least 4x speedup on the GHC machines. We will attempt a version for CPUs with multiple cores using OpenMP, and also a version for GPUs using CUDA.

URL: https://www.andrew.cmu.edu/user/jsliu2/418projectproposal.pdf

BACKGROUND:

Seam Carving is an image resizing algorithm that works by removing seams of least importance. A vertical seam is defined as a path from the top to bottom of an image of adjacent pixels such that there is exactly one pixel per row on the path. A pixel is considered adjacent from another pixel if the pixels are in adjacent rows and their column indices differ by at most 1. For this illustration, we will be working with vertical seams.

Seam Carving is considered as a better alternative to cropping or scaling an image in order to resize it because it better preserves the field of vision of the original image. In this series of examples taken from the 15-210 DP Lab writeup, the rightmost Seam-Carved image does a much better job preserving the details of the rocks while maintaining a view of the water.



The intuitive directions for parallelism would be to parallelize the work across columns when vertical seams are being removed. This alone, however, is unlikely to achieve the desired speedup, since this still requires synchronization for each row, and for a given row, there is not enough work to be split. Thus, we will search for alternative ways to partition the tasks so that each task has a larger granularity. Another natural extension is to remove multiple seams at once in parallel. However, it is nontrivial to identify nonintersecting seams and, furthermore, the removal of each seam depends on what was removed previously, so there will be challenges involving balancing between achieving more parallelism and more accuracy.

THE CHALLENGE: Describe why the problem is challenging. What aspects of the problem might make it difficult to parallelize? In other words, what to you hope to learn by doing the project?

- Describe the workload: what are the dependencies, what are its memory access characteristics? (is there locality? is there a high communication to computation ratio?), is there divergent execution?
- Describe constraints: What are the properties of the system that make mapping the workload to it challenging?

In the DP solution to making a valid Seam Carver via vertical seams, the pixel energy matrix is traversed one row at a time such that the computation for each row depends on having finished the computation for the row directly above it. This means that the computation for each row has to be synchronized before the computation for the next row can be started in the optimal/sequential solution. This presents our main challenge: With so many row dependencies, only parallelizing across the columns will not be enough to overcome the cost of synchronizing each row, especially when each pixel only has to look at the 3 pixels above it, which is a very small amount of work. We will try to find ways to address this issue. Additionally, we will probably have to investigate how to balance between optimally removing the seam of lowest energy and achieving greater speedup. One idea we are entertaining is to remove multiple seams in parallel. For example, if we wanted to remove 2 vertical seams in parallel then we would divide the image in half and remove a seam from each half such that the seams don't touch the other half. Another challenge is what to do after we have identified the seam to remove- the seam to remove could be spread across the whole image so we have to figure out how to efficiently copy the remaining pixels into a new image in parallel.

In terms of memory access characteristics, there is a fair amount of spatial locality because each pixel accesses a contiguous set of 3 pixels in the row above and adjacent pixels in a row will access 2 of the same pixels in the row above. There should be very little divergent execution because the only edge cases that do not access 3 pixels are the leftmost/rightmost pixel in each row.

RESOURCES: Describe the resources (type of computers, starter code, etc.) you will use. What code base will you start from? Are you starting from scratch or using an existing piece of code? Is there a book or paper that you are using as a reference (if so, provide a citation)? Are there any other resources you need, but haven't figured out how to obtain yet? Could you benefit from access to any special machines?

- We will start by using this person's public Seam Carver code as our starter code: https://github.com/dwxiao/seam-carving/blob/master/seam-carving.cpp
- Our initial plan is to follow the DP version of the Seam Carving algorithm that's implemented already in the starter code as well as described in this link: https://en.wikipedia.org/wiki/Seam_carving#:~:text=Seam%20carving%20
- We will use the GHC machines and request use of the PSC machines to see how our implementation performs at higher core counts.
- We will also be referencing how to use CUDA from our asst2 code and how to use OpenMP from our asst3 code

GOALS AND DELIVERABLES: Describe the deliverables or goals of your project. *This is by far the most important section of the proposal!*

- Separate your goals into what you PLAN TO ACHIEVE (what you believe you must get done to have a successful project and get the grade you expect) and an extra goal or two that you HOPE TO ACHIEVE if the project goes really well and you get ahead of schedule, as well as goals in case the work goes more slowly. It may not be possible to state precise performance goals at this time, but we encourage you be as precise as possible. If you do state a goal, give some justification of why you think you can achieve it. (e.g., I hope to speed up my starter code 10x, because if I did it would run in real-time.)
- If applicable, describe the demo you plan to show at the poster session (Will it be an interactive demo? Will you show an output of the program that is really neat? Will you show speedup graphs?). Specifically, what will you show us that will demonstrate you did a good job?
- If your project is an analysis project, what are you hoping to learn about the workload or system being studied? What question(s) do you plan to answer in your analysis?
- Systems project proposals should describe what the system will be capable of and what performance is hoped to be achieved.

Plan to Achieve

- A parallelizable version of the DP algorithm implementation of a Seam Carver that carves vertical seams. The initial version will not be focused on runtime speedup, just getting a version that behaves correctly when run on multiple processors.
  We think that this is achievable because this is essentially taking the sequential code and distributing the work among multiple processors, so maintaining correctness should not be super hard.
- Some sort of correctness verifier that we can use to test the parallel version of Seam Carver. We can assess the quality of an image both via human verification and the resultant energy of the image.
  We think that this is achievable because this will be a separate check from the runtime tests so we don't have to worry about making this efficient. We can simply take in a seam from the sequential version and verify that it is the same seam as found in the parallel version

- A version of the DP implementation of a Seam Carver that carves vertical seams where the speedup from the sequential version is 4x.
  If we assume that we cannot achieve perfect speedup and the GHC machines and the GHC machines can employ at most 8 processors, then we think that achieving at least half of 8x speedup (4x) is reasonable.

Hope to Achieve

- We can try to also parallelize Seam Carving with OpenMP. This seems feasible because we have worked with OpenMP for assignment 3 so we have experience and reference material. Moreover, if we've already hashed out how to effectively parallelize by now, then changing syntax should not be too hard.

Demo at poster session: We will be presenting speedup graphs with a sequential version as a baseline. We will also provide images that have been Seam Carved to various degrees of resizing as a more visual representation of what our project does.
We hope to determine whether multi-core CPUs or GPUs are best suited for parallel seam carving, and we plan to understand the bottlenecks of parallelizing this algorithm and what limits its performance.

PLATFORM CHOICE: Describe why the platform (computer and/or language) you have chosen is a good one for your needs. Why does it make sense to use this parallel system for the workload you have chosen?

We will be choosing to work in C++ because we have worked with C++ for all of our prior assignments for this class and are familiar with how to apply parallelism using techniques such as CUDA. CUDA makes sense for this workload because we don't need very much communication between pixels and we only need to synchronize, so CUDA would be great at dividing up this labor. We will be using the GHC machines to test our speedup.

SCHEDULE: Produce a schedule for your project. **Your schedule should have at least one item to do per week.** List what you plan to get done each week from now until the parallelism competition in order to meet your project goals. Keep in mind that due to other classes, you'll have more time to work some weeks than others (work that into the schedule). You will need to re-evaluate your progress at the end of each week and update this schedule accordingly. Note the intermediate milestone deadline is April 16th. In your schedule we encourage you to be precise as precise as possible. It's often helpful to work backward in time from your deliverables and goals, writing down all the little things you'll need to do (establish the dependencies).

Week	Deadline	Task(s)
1	Wed March 27	- submit project proposal
2	Tues April 2	- update/finalize project proposal based on feedback from submitted proposal

3	Tues April 9	- modify starter code to get a working sequential version we can write parallel code off of - establish how we will parallelize finding multiple seams
4	Tues April 16	- working parallel version of sequential code - identify bottlenecks - run experiments to compare various ideas we had - understand the tradeoff with finding multiple seams and the image quality
5	Tues April 23	- try to get as much speedup as possible on the GHC machines - test on PSC machines to obtain speedups on higher core counts
6	Sun May 5	- finish final report - finish poster