Project 1 – Seam Carving

COMP 6651 – Winter 2018

Dr. Tiberiu Popa

Introduction

Image retargeting is the process of pasting the content of an image of certain size to a canvas of a different size. Figure 1 illustrates this process: the goal is to take the content of Image 1a) and paste it to a canvas that is shorter along the x axis. Common methods include non-uniform scaling (Figure 1b) or cropping (Figure 1c). However, both have their shortcomings. Non-uniform scaling may distort the proportions of the objects in the scene and cropping might remove important content. A better solution is seam-carving as shown in Figure 1d).

In this project, you are asked to implement a simple seam carving technique using dynamic programming and the OpenCV library.

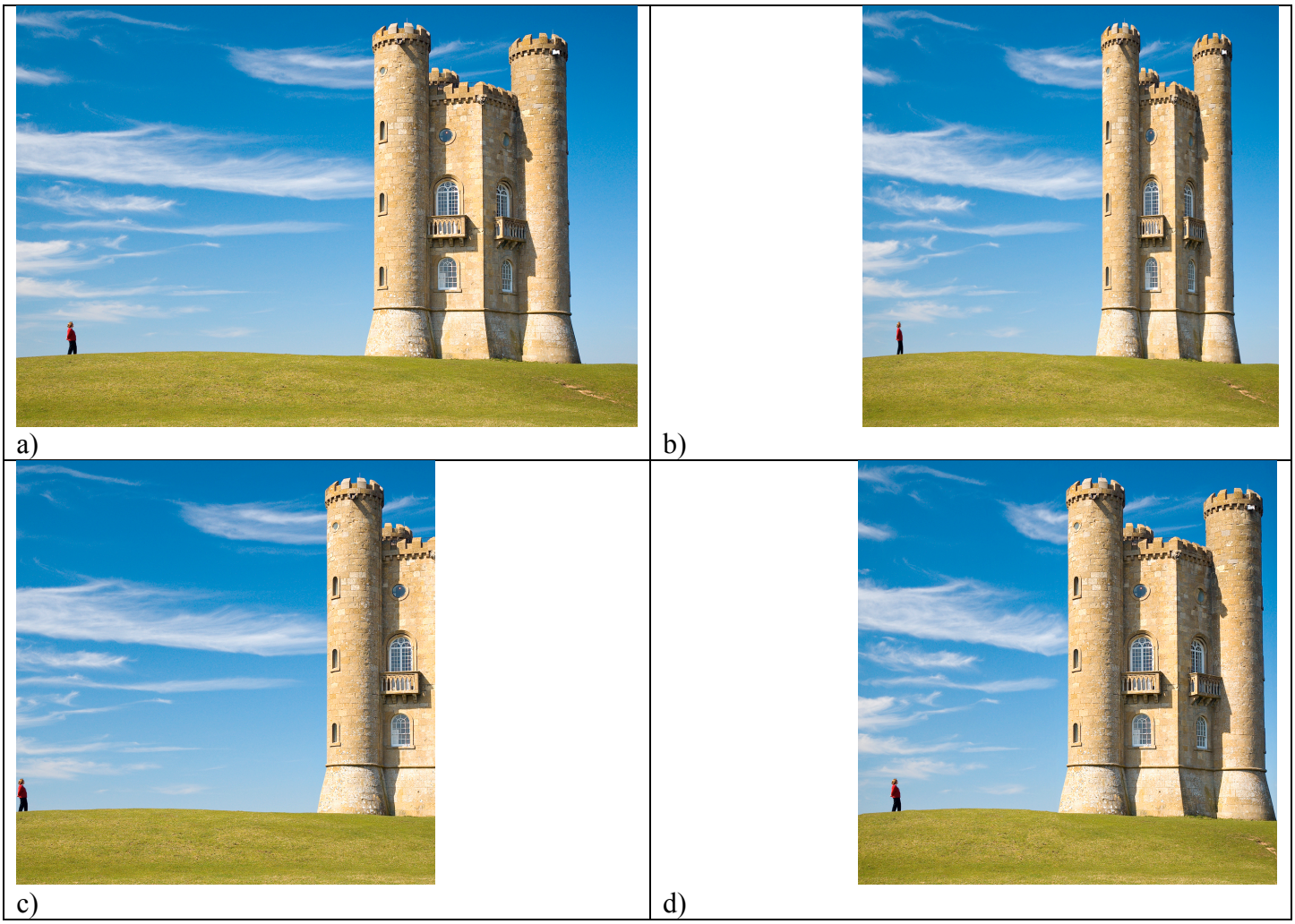

Figure 1. Retargeting images examples (image taken from https://en.wikipedia.org/wiki/Seam_carving). a) original image. b) Retargeting by non-uniform scaling. c) Retargeting by cropping. d) Retargeting using seam carving.

Seam Carving

In broad terms the seam carving algorithm uses horizontal or vertical seams (Figure 2(red)) to iteratively reduce the size of the image. A seam is a pixel thick cut through the image that stretches either from left to right or from up to down (depending on the orientation of the seam). Removing the seam reduces the image by one unit. For example, a horizontal seam will reduce the size of the image by one row, and a vertical seam by one column.

Figure 2. Horizontal and vertical seam (rendered in red) from the seam carving algorithm. Image taken from [1]

The main idea is to find the seams that cut through the non-salient content of the image. In the castle example, we want to generate seams that do not remove details form the castle and the person in the image. Such seams are illustrated in Figure 3. A good seam basically cuts through pixels that are similar with each other. In other words, a good seam is a seam that has small variations in color along the seam. Finding such optimal seams can be done using a dynamic programming algorithm.

Seam carving is a very popular algorithm [1] despite being relatively new, used in most image processing packages. You can find plenty of resources on the web, an excellent one that explains the algorithm in detail is this: http://www.wikiwand.com/en/Seam_carving - /overview **Note that you do not have to implement all the features described in the original paper, the specifications section provides more details on what parts of the algorithm you need to implement.**

Figure 3. Valid seams that avoid removing salient details from the castle and the person. (image taken from https://en.wikipedia.org/wiki/Seam_carving)

Specifications

The program has 3 arguments: an input image, width of the new target, height of the new target, output image. Your program has to retarget the input image to a new image of given sizes and write it into a file denoted by the last argument. The minimum size of the target image is 2x2 pixels.

Example: *./sc castle.png 300 200 castle2.png*
Your program (has to be called *sc* for our automatic testing) has to generate a new image of size 300x200 that retargets the image in castle.png and saves it in the file castle2.png

You need to implement the dynamic algorithm for seam carving described in the previous section. You do not need to solve the general case, you can assume that the new dimensions of your image are not larger than the original image (i.e. you only have to scale down and not up). For the energy function for the seam carving, you can use the simplest version that still provides adequate results.

What is given

You are given a codebase that does a trivial seam carving: always selects as the current seam a horizontal or vertical line (as needed) at the middle of the image and removes it iteratively to decrease the size of the output image. This is of course not the desired behavior, but this codebase is just a platform for your solution as well as a jump start with OpenCV.

The codebase has a src folder that contains three files: main.cpp, sc.h and sc.cpp. The *main.cpp* contains the main function and the initial parsing of the arguments. It then calls the *seam_carving* function with the appropriate parameters and after that it saves the files in the appropriate folders and also it displays them on the screen. The *sc.h* contains the declaration of the *seam_carving* function. The file *sc.cpp* contains the implementation of the *seam_carving* function as well as other useful functions. In the version provided, the *seam_carving* function calls a dummy *seam_carving*_trivial function with the functionality mentioned above.

You will need to implement the seam_carving function given the correct implementation describe above. You should be able to do so only changing the file *sc.cpp,* but you are allowed to change any part of the code.

The codebase provided is configured and compiled with cmake. The only other dependency is opencv. The project is configured to run on any machine that has a properly installed opencv, but it was tested only on the linux machines in the labs as well as on a Mac OS X Sierra.

In the linux lab you need to run first the command *module load opencv*. This command will mount the folder where OpenCV is installed. Read carefully the README file provided with the code to compile and run the project. After compilation te executable will be called sc and that the executable is stored in the build folder and it is not mixed with the source files that are stored in the *src* folder.

For testing you can select any images from the web. Nevertheless, we added the castle image shown above in the build folder.

Submission

You have to implement your program in C/C++ using OpenCV and the codebase provided. You are allowed to change the codebase, but the following steps cannot be changed: 1) the project must be configured using only the command cmake../ from the build folder. 2) the project must be compilable using only the command make from the build folder. 3) the project must be run using the specifications from the previous section, including the name of the project. As the compilation and execution is automated, failure to meet these requirements will result in a grade of 0.

Unlike the exercises, the project you have to make a zip file of the entire project and submit it using the EAS system under "**Project 1**". The due-date is **March 16th, 5pm.** The deadline is soft in that we accept late submissions at a penalty of 20% (of the total marks) for every 24-hour period.

This exercise is individual.

Unlike the exercises, the projects are fairly complex and require a lot of work, therefore please start working on it early.

Originality and Plagiarism

Seam carving algorithm is a well-known algorithm, whose code undoubtedly can be found on the internet. This is a reminder to everyone that everything you submit must be your original work and you are expected to be able to explain the code in detail if such a request is made by the teaching assistants or the instructor.

Evaluation and Testing

The code is compiled and ran automatically on a number of images. However, the evaluation will be done by a human visually. You will receive a code of 0 if the code does not build and execute as specified. You may be asked to explain your code to the teaching assistant or the instructor. This may lead to a decrease in your overall grade for this exercise.

References

[1] Avidan, S., & Shamir, A. (2007, August). Seam carving for content-aware image resizing. In *ACM Transactions on graphics (TOG)* (Vol. 26, No. 3, p. 10). ACM.