TJHSST Computer Systems Lab Senior Reasearch Project Dynamic Image Resizing Quarter 3 Paper

Patrick Elliott

April 4, 2008

1 Abstract

The goal of this project is to be able to resize an image without distorting any important aspects of the image. Commons methods of resizing, including cropping and scaling, remove or distort some of the image and are thus undesirable. By finding the least important pixels and removing them, this dynamic resizing can be possible. These can be found by finding the change of intensity of each pixel to the next and taking away the ones with a very low change. Using this method, humans should be unable to tell if an image has been altered.

2 Introduction

Currently on the web, there is such a thing as dynamic text formatting. For instance, when you resize a web browser window, the text in it will adjust itself to fit inside the window while still being readable. There is nothing like this for images however. My goal in this project is to be able to change the dimensions of an image without losing important content, such as the dimensions of the focus of these pictures.

3 Background

Edge detection is being researched heavily in modern times. Many teams of researchers are trying to allow computers to see and identify objects. But there is also much research being conducted concerning editing modifying images. There is one project called PhotoSynth that is trying to take a large amount of images from the web, and from them, create a 3D model of whatever the images are of. There is also another project that is very similar to what I am attempting to do, although I have some ideas for my project that they have not yet implemented.

4 Development

I will be using C for all of my programming. My approach to content-aware resizing is to remove pixels in a judicious manner. Therefor, the question is how to chose the pixels to be removed? Intuitively, my goal is to remove unnoticeable pixels that blend with their surroundings. This leads to the following simple energy function: Each pixel's value is the difference in intensity with the pixel's neighbors. Given an energy function, assume we need to reduce the image width. One can think of several strategies to achieve this. For instance, an optimal strategy to preserve energy (i.e., keep pixels with high energy value) would be to remove the pixels with lowest energy in ascending order. This destroys the rectangular shape of the image, because we may remove a different number of pixels from each row. If we want to prevent the image from breaking we can remove an equal number of low energy pixels from every row. This preserves the rectangular shape of the image but destroys the image content by creating a zigzag effect. To preserve both the shape and the visual coherence of the image we can use auto-cropping. That is, look for a sub-window, the size of the target image, that contains the highest energy. However, if two important objects are on either side of the target image, this wont achieve the desired effect. Another possible strategy somewhat between removing pixels and cropping is to remove whole columns with the lowest energy. Still, artifacts might appear in the resulting image. Therefore, we need a resizing operator that will be less restrictive than cropping or column removal, but can preserve the image content better than single pixel removals. This leads to my strategy of seam carving. Rather than removing scattered pixels or entire columns, my program removes seams

(vertical paths of pixels within one column of the pixel above it) of the lowest intensity. The optimal seam can be found using dynamic programming. The first step is to traverse the image from the second row to the last row and compute the cumulative minimum energy M for all possible connected seams for each entry (i, j): M(i, j) = e(i, j) + min(M(i-1, j-1),M(i-1, j),M(i-1, j+1)) where the function e is the intensity of the given pixel. At the end of this process, the minimum value of the last row in M will indicate the end of the minimal connected vertical seam. Hence, in the second step we backtrack from this minimum entry on M to find the path of the optimal seam. The definition of M for horizontal seams is similar. To enlarge an image we insert new artificial seams to the image. Hence, to enlarge the size of an image I by one we compute the optimal vertical (horizontal) seam s on I and duplicate the pixels of s by averaging them with their left and right neighbors (top and bottom in the horizontal case).

5 Results

As of now, the program is capable of shrinking the image in both the horizontal and vertical directions and enlarging in the horizontal direction. With the newest method of finding the optimal pixels to remove (the cumulative sums of the gradient image) the final images are usually very realistic. It is difficult to notice that it has been modified at first glance, which is the goal of the project.