

Technion – Israel Institute of Technology

Department of Electrical Engineering

Laboratory of Computer Graphics and Multimedia

Project Report for

Content-Aware Image Resizing Using Seam Carving

Author: Tsur Amit – 038164927

Supervisor: Varod Danny, B.Sc.

Table of Contents

[Abstract 3](#_Toc233890492)

[Introduction 4](#_Toc233890493)

[Project Goals 5](#_Toc233890494)

[Algorithm 6](#_Toc233890495)

[Pixel Energy 6](#_Toc233890496)

[Seam 6](#_Toc233890497)

# Abstract

Image resizing is mainly known as cropping or scaling of images. These methods of resizing are not content-aware; they treat each element of the image the same as the other. Now more than ever, with the popularity of the Internet and mobile devices and with the power of modern CPU/GPU, effective image resizing is getting more and more important and necessary.

This project tries to implement a method of resizing an image in a content-aware fashion. Content-aware shrinks/enlarges the image by removing/inserting the least interesting parts of an image, thus resizing an image with regards to its content.

# Notes

Throughout the paper, there might be references to actions in horizontal and vertical direction; if an elaboration in one direction is missing, the reader can assume that the same action can be done in respect to the other direction in a symmetric manner. This is also true for enlargement and shrinking.

# Introduction

Seam Carving is achieved by firstly assigning a weight to each element of the image. This can be done in various methods and depends on the type of image. Some methods can perform better than others on specific images. The next step is to calculate a cumulative map of weights, used to determine how important/interesting each element of the image is.

We then add or remove to the image the least interesting elements of it, hence keeping the affect on the content at the minimum.

This project focuses on implementing a fast and efficient content-aware image resizing application based on the article by Shai Avidan and Ariel Shamir.

# Project Goals

* Implement Shamir and Ariel’s algorithm.
* Build a comfortable GUI to use the implementation.
* Test its many applications.
* Try and analyze different functions.
* Optimize it further to achieve better performance using various methods.

# Algorithm

## Pixel Energy

As mentioned before, each element (pixel) of the image must have some numerical value, indication of how ‘interesting’ it is. The numerical value is called *Energy*, and in our application we’ve chosen to use the pixel’s luminance. Luminance is a photometric measure of the luminous intensity. We’ve used *Luma*, which is the representation of luminance in a video monitor. Luminance of a pixel is achieved with the following:

Y = 0.2126 R + 0.7152 G + 0.0722 B

We’ve chosen luminance because it has proven itself as a good measure of energy; it normalizes images in a satisfying way, which allowed us to see a clear energy map of the image using the different energy functions (will be discussed later). Another way to measure the energy is by using grayscale intensity, which has shown similar results.

There are many representations of the pixel’s energy. Although they have an impact, we settled for luminance and focused on the other aspects of the program.

As an image is loaded, the energy is calculated for each pixel and stored in the energy map.

## Seam

Seam is an 8-connected path of pixels. A vertical seam is a seam from top to bottom which contains exactly one pixel from each row of the image, and more specifically, for an image  of size  we define a vertical seam to be:



Where x is a mapping .

## Cumulative Energy

After calculating the energy, we find the cost of the minimal seam. We define the cost of a seam:



In order to fine the minimal seam cost, we use dynamic programming. We traverse the image from the second row to the last, and calculate the cumulative minimum energy  for all possible connected seams for each entry :



After calculated, the cumulative energy is saved in the cumulative energy map.

## Optimal Seam

After the completion of the process, we scan the last row for the minimal value. The minimal value is the cost of the minimal (optimal) seam. We then backtrack from the minimal entry in an 8-connected manner to the minimal neighbor until we reach the top of the image again. The minimal path defines the minimal seam.

## Index Map

Vertical index map of image  of size  is a sized  integer array.

For any indices , the following holds:

 The pixel in the image is removed in the  seam removal

The index map is used in order to efficiently resize the image. The usage will be explained throughout the paper.

## Seam Removal / Addition

In order to shrink (enlarge) an image of size  to size  where  (), we remove (add) the first  () optimal seams.