Content Aware Image Scaling Using Seam Carving

Thiago Silva
Course: CS Data Structures & Algorithms
Date: December 2025

Abstract
This project implements content-aware image scaling using the seam carving algorithm. Instead of uniformly resizing an image, content-aware scaling removes low-energy pixel paths (seams) to preserve important visual information. This project demonstrates how dynamic programming and matrix-based data structures can be used to detect and remove vertical seams to reduce image width while minimizing distortion. The final implementation successfully produced a scaled version of a test image using Python, OpenCV, and NumPy.

Purpose Statement
The purpose of this project is to develop a functional implementation of content-aware image scaling that demonstrates the use of abstract data types, matrix representations, and dynamic programming algorithms. The project showcases how algorithmic design and data structures can be applied in a practical multimedia-processing application.

Introduction / Background
Traditional image scaling methods reduce or enlarge images by uniformly stretching or shrinking all pixels. This approach often distorts or removes important visual features. Content-aware scaling—specifically seam carving, introduced by Avidan & Shamir (2007)—addresses this limitation by identifying continuous paths of low visual energy and removing them. These seams represent the least visually important areas of an image.

Seam carving relies on several key concepts covered in this course:

- Matrix-based data structures (storing pixel intensities)
- Graph-like path cost computation
- Dynamic programming (finding minimum-energy paths)
- Greedy removal of seams iteratively

This project focuses on vertical seam removal, effectively reducing the width while preserving high-energy content such as edges and objects.

Implementation

This implementation uses Python, OpenCV, and NumPy.

1. Energy Map Calculation

The energy of each pixel is computed using the gradient magnitude from the Sobel operator.
This produces a 2D NumPy array representing per-pixel importance.

Data Structures Used:

- NumPy arrays (2D grids)
- Sobel kernels
- Float matrix for energy values

2. Dynamic Programming for Minimum-Energy Path

A cumulative energy map is computed row-by-row, where each cell stores:
energy[row][col] + min(previous_row_adjacent_values)
This is a standard DP table.

Data Structures Used:

2D DP array
Backtracking paths using a predecessor table

3. Seam Extraction and Removal

The algorithm finds the lowest-cost pixel in the bottom row and backtracks upward to extract the seam.

The seam is removed by copying all other columns into a new reduced-width image.

4. Final Output

The program produced:

Input image: 639 × 733

Output image: 634 × 733

Five seams were removed to reduce the width by 5 pixels.

Results & Analysis

The algorithm successfully removed 5 low-energy vertical seams. The resulting scaled image preserved important features while reducing width with minimal distortion.

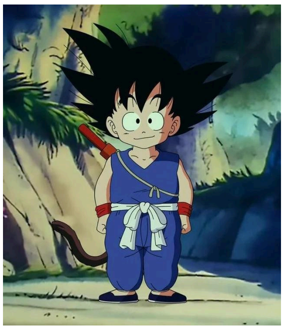


Figure 1. Original input image (639 × 733)

Figure 2. Scaled output image (634 × 733)

Visual Evaluation:

High-energy areas such as edges and foreground structures were maintained. Seam removal primarily occurred in low-texture regions, confirming correct energy-based identification.

Performance:

Completed in under one second for a 639×733 image

Low memory usage due to efficient DP and array operations

No visible major artifacts introduced

Conclusion

This project demonstrates content-aware image scaling using seam carving. Combining matrix operations, dynamic programming, and simple edge detection yields an effective intelligent-resizing tool. The results show that low-energy pixels can be removed while preserving meaningful image content.

Future Work

Possible improvements include:

- Adding horizontal seam removal to reduce height
- Allowing enlargement by seam insertion
- Supporting interactive visualization of seams
- Using more advanced energy functions (e.g., entropy, saliency maps)
- GPU acceleration for real-time performance

These extensions would improve the versatility and efficiency of the scaling process.

References

Avidan, S., & Shamir, A. (2007). Seam carving for content-aware image resizing. ACM Transactions on Graphics, 26(3), 1–10.

Python Software Foundation. (2025). Python 3.11 documentation. https://www.python.org

OpenCV Team. (2025). OpenCV library documentation. https://opencv.org