CS7.404 : Digital Image Processing
Enhanced Video Retargeting Using Graph Cuts and Forward Energy Seam Carving

Vyakhya Gupta
2022101104

Prakhar Jain
2022121008

November 30, 2024

Note : The code is available here

Abstract

This work explores the implementation of seam carving, a content-aware resizing technique for images and videos, enabling dimension adjustments while preserving essential visual content. Seam carving identifies and manipulates paths of least energy—called seams—using an energy function, facilitating content-aware resizing for both reduction and expansion. The approach has been applied to images using dynamic programming and extended to videos by leveraging graph cuts for spatial and temporal coherence. To improve visual quality, forward energy minimization is used, reducing artifacts during resizing. The results demonstrate the versatility of seam carving for tasks such as image retargeting, video resizing, content enhancement, and the creation of multi-size adaptable media. This method provides an effective solution for dynamically adapting media to various display devices and layouts.

Introduction

The diversity of modern display devices, ranging from smartphones to high-resolution monitors, creates a growing need for adaptable digital media. Unlike text, which can dynamically adjust to fit varying layouts, images and videos often remain static in size and aspect ratio. Traditional resizing methods, such as cropping or uniform scaling, are limited as they disregard content significance, potentially leading to loss of important visual elements or noticeable distortions.

Seam carving [1, 2, 3] addresses these challenges by introducing a content-aware resizing approach. Seams are defined as connected paths of pixels with minimal energy, determined by an energy function. By selectively removing or inserting these seams, dimensions can be adjusted while preserving the most critical visual content. This method has been applied to images using dynamic programming to identify and manipulate seams effectively.

For videos, resizing presents additional challenges, such as ensuring temporal coherence across frames. To overcome this, seam carving has been extended to 3D space-time volumes using graph

cuts, which identify connected, monotonic manifolds. Forward energy minimization is also incorporated, addressing the introduction of artifacts and preserving both spatial and temporal features.

Applications of seam carving include aspect ratio adjustments, multi-size image creation, and content-aware video retargeting. These capabilities enable media to adapt dynamically to a variety of devices and layouts while maintaining visual integrity. This approach highlights the potential of content-aware resizing as a robust solution for meeting modern media adaptability requirements.

Related Work

In many scenarios, obtaining an image with specific dimensions requires manual resizing. The required dimensions are often rigid, leading to compromises in image content. Common methods like scaling and cropping are typically employed for resizing. However, these approaches can distort the image's contents—either altering the original appearance or losing important surrounding information. When applied to images containing multiple objects, the quality often degrades, and critical details may be lost.

Setlur et al. [4] proposed an advanced technique called Automatic Image Retargeting. This method involves segmenting the image into regions of interest, identifying and removing specific regions, filling gaps, resizing the remaining areas, and reinserting important regions to produce the final output. While effective at preserving multiple key features and producing aesthetically pleasing results, this approach is computationally intensive due to its multi-step process.

Gal et al. [5] introduced a method that warps an image into an arbitrary shape while preserving user-defined features. By utilizing a specialized Laplacian editing formulation, the method enforces similarity constraints on specific areas. However, as these constraints propagate globally, not all can be satisfied simultaneously, leading to trade-offs in the final output.

Grundmann et al. [6] proposed a novel video retargeting algorithm employing discontinuous seam carving in both spatial and temporal dimensions. Their approach leverages an appearance-based temporal coherence model, optimizing differences between the retargeted and temporally coherent frames. Additionally, the algorithm introduces piecewise spatial seams to improve spatial detail retention, outperforming traditional methods that focus solely on minimizing color differences.

Approach

Seam Carving

Seam carving is a technique used to adjust the aspect ratio or size of an image while maintaining its essential content. It works by removing the pixels with the lowest energy, thereby preserving the most important features of the image. However, this method may distort the rectangular shape of the image. To prevent such artifacts, a seam, which is a path of connected pixels, is removed either vertically or horizontally. When shrinking an image, the lowest energy seam is continuously removed, and the process is repeated until the target dimensions are achieved.

Image Retargeting

A vertical seam consists of an 8-connected path of pixels extending from the top to the bottom of the image, with one pixel selected from each row. Similarly, a horizontal seam is an 8-connected path of pixels running from left to right. To identify the best seam for removal, we begin by creating an energy map of the image. The gradient method using the Sobel filter provides particularly effective results for this purpose. The energy function is defined as:

$$e(\mathbf{I}) = \left|\frac{\partial \mathbf{I}}{\partial x}\right| + \left|\frac{\partial \mathbf{I}}{\partial y}\right|$$

The optimal seam can then be found using dynamic programming. The first step involves traversing the image from the second to the last row, computing the cumulative minimum energy M for all possible connected seams at each position (i, j). The recursive formula for calculating $M(i, j)$ is:

$$M(i,j) = e(i,j) + \min \begin{cases} M(i-1, j-1) \\ M(i-1, j) \\ M(i-1, j+1) \end{cases}$$

After evaluating all possible seams, the minimal value in the last row of M indicates the endpoint of the minimal vertical seam. If multiple minimal values are found, the entry with the smallest j-coordinate is selected. The second step is to backtrack from M to trace the optimal path. A similar process is followed for horizontal seams.

Forward Energy

The original algorithm introduces artifacts into the image as it neglects the energy inserted into the retargeted image. To address this, a new criterion is introduced for selecting the optimal seam. The idea is to consider the resulting image after seam removal. When a seam is removed, the neighboring pixels on either side of the seam merge, forming new "pixel-edges." Since this removal impacts both the image and its energy in a local neighborhood, it is sufficient to examine only a small region around the removed pixel. The cost of these new "pixel-edges" is computed based on the forward differences between the pixels that will become new neighbors after the seam removal. There are three possible cases based on the seam direction: (a) the left pixel, (b) the vertical pixel, and (c) the right pixel (as shown in Figure 1). The step costs for pixel $p_{i,j}$ using forward energy are given by:

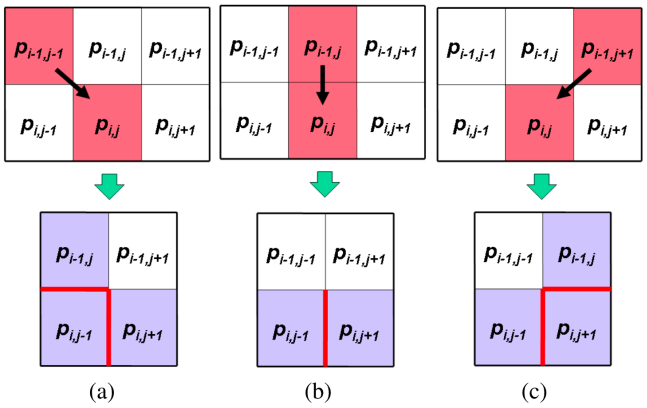

Figure 1: Neighbourhood changes after seam removal

Left and Right neighbours:

$$+LR = |I(i, j+1) - I(i, j-1)|$$

Left and Up neighbours (upward vertical)

$$+LU = |I(i, j-1) - I(i+1, j)|$$

Left and Up neighbours (downward vertical)

$$-LU = |I(i, j-1) - I(i-1, j)|$$

Figure 2: Neighboring differences for Left-Right and Left-Up pairs.

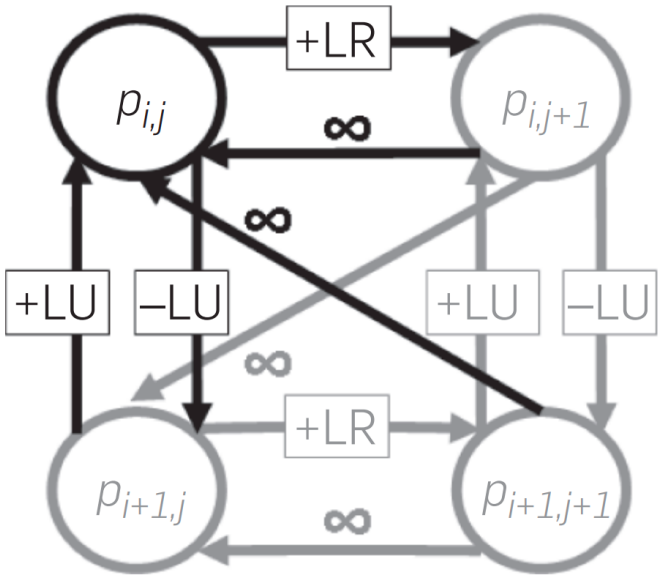

Figure 3: Graph construction with forward energy

$$
\begin{aligned}
Left : C_L(i,j) &= |I(i,j+1) - I(i,j-1)| + |I(i-1,j) - I(i,j-1)| \\
Vertical : C_V(i,j) &= |I(i,j+1) - I(i,j-1)| \\
Right : C_R(i,j) &= |I(i,j+1) - I(i,j-1)| + |I(i-1,j) - I(i,j+1)|
\end{aligned}
$$

The new step costs are then incorporated into the cumulative cost matrix M calculation using dynamic programming, as described in the subsequent work [2]. The seam with the minimum cost is then removed from the image.

Video Retargeing

Unlike static images, the dynamic programming approach cannot be directly applied to videos for seam carving due to the need to preserve temporal coherence across frames. To address this, Rubinstein [2] proposes a graph-based method leveraging the minimum-cut-maximum-flow algorithm. This approach represents the video as a graph, where each pixel is treated as a node, and energy values—calculated using forward energy to account for object boundaries—define the edges between neighboring pixels. This ensures the seamless resizing of frames while maintaining temporal consistency.

Graph Cut

The graph cut approach defines an S/T cut C as a partition of the nodes in a graph into two disjoint subsets, S and T, such that the source node $s \in S$ and the sink node $t \in T$. The cost of a cut $C = \{S, T\}$ is determined by the sum of the costs of the boundary arcs (p, q), where $p \in S$ and $q \in T$. A seam is derived from the cut by consistently selecting pixels to the left of the cut arcs. The optimal seam is identified as the minimum cut, having the lowest cost among all valid cuts, while adhering to the following constraints:

1. **Monotonicity**: The seam must include exactly one pixel in each row (or column, for horizontal seams).
2. **Connectivity**: The pixels forming the seam must be connected.

This formulation ensures that the seam constraints are satisfied, as detailed in the proof provided in the referenced work.

3D Graph Cut

For videos, we extend this concept to a 3D graph constructed from an $X \times Y \times T$ video cube, where X and Y represent spatial dimensions and T represents the temporal dimension. To compute a vertical seam, we consider $X \times T$ planes and employ a similar graph construction as in $X \times Y$, with backward diagonal infinity arcs ensuring connectivity. The source node is connected to all leftmost columns across frames, while the sink node is connected to all rightmost columns. For horizontal seams, the source node is connected to the topmost rows and the sink to the bottommost rows across all frames. The resulting cut defines a manifold in the 3D space, maintaining monotonicity and temporal coherence. The graph cut produces a one-dimensional connected seam in each frame, achieving global optimality across the video cube.

Look-Ahead Energy

Rubinstein [2] highlights that constructing a full 3D video cube, where each voxel corresponds to a node, is computationally impractical for high-resolution inputs due to the quadratic growth in graph complexity. To address this, we propose an approach that avoids constructing the full 3D graph. Instead, we maintain temporal coherence by introducing a look-ahead energy model.

In this method, the seam energy depends on the gradients of the current frame as well as gradients from subsequent k frames. For a given frame i, the new energy values for neighboring pixels are calculated as a weighted linear combination of gradients from the current and next four frames:

$$
\begin{aligned}
+LR_i &= \alpha_i LR_i + \alpha_{i+1} LR_{i+1} + \cdots + \alpha_{i+4} LR_{i+4}, \\
+LU_i &= \alpha_i LU_i + \alpha_{i+1} LU_{i+1} + \cdots + \alpha_{i+4} LU_{i+4}, \\
-LU_i &= \alpha_i(-LU_i) + \alpha_{i+1}(-LU_{i+1}) + \cdots + \alpha_{i+4}(-LU_{i+4}),
\end{aligned}
$$

subject to the constraint:

$$
\sum_{f=i}^{i+4} \alpha_f = 1.
$$

This approach balances the influence of multiple frames while preserving temporal consistency, enabling efficient seam computation without requiring a full 3D graph representation.

Experiments

We implement the above methods utilizing Python and NumPy for flexibility and ease of development. For the graph-cut algorithm, we adapt the max-flow algorithm using the `maxflow` library in Python, closely following the logic provided by [1] and [3].

To enhance performance in video retargeting, we parallelize the seam removal process using Python's `concurrent.futures.ProcessPoolExecutor`. This approach leverages multiple processes to divide the computation across available CPU cores, enabling efficient parallel execution. As illustrated in Figure 6, utilizing parallel processing significantly reduces the time required for seam removal. Optimally, we use 6 processes in video retargeting.

In our experiments, the time taken to remove one vertical seam in the airplane video video using the 3D graph-cut method is approximately **3.5 minutes**, whereas our approach with multi-threading requires around **40 seconds**. This demonstrates the effectiveness of our parallelized implementation in achieving substantial time savings.

Images

For images, we conducted experiments involving both reduction and expansion on various test images to evaluate the performance and quality of the seam carving algorithm.

Figure 4: Original Image before Seam Carving

Figure 5: Resulting Image after applying Seam Carving using DP

Videos

For videos, we successfully implement the 3D graph cut on the video cube using our Python-based approach, leveraging the `maxflow` library for the graph-cut algorithm. Results of seam removal are shown and discussed in the next section. The experiment for video retargeting compares the results generated by our parallelized implementation with those produced by the original approach, and varying α values.

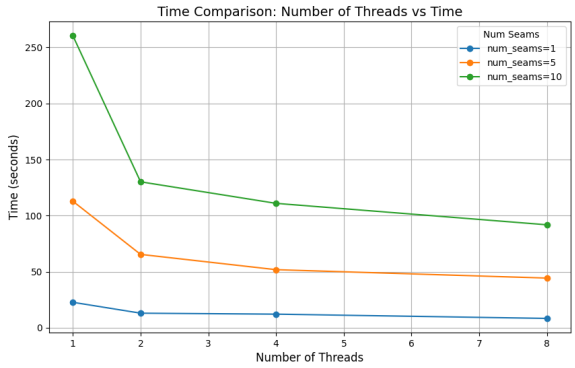

Figure 6: Time comparison between different numbers of threads and seams. Each line represents a different `num_seams` value, showing the impact of thread count on execution time.

We experimented with different α values to evaluate their impact on the seam carving results. The following configurations were used for α values:

- **Uniform Distribution:** $\alpha_1 = 0.2, \alpha_2 = 0.2, \alpha_3 = 0.2, \alpha_4 = 0.2, \alpha_5 = 0.2$
- **Temporal Decay:** $\alpha_1 = 0.4, \alpha_2 = 0.2, \alpha_3 = 0.15, \alpha_4 = 0.15, \alpha_5 = 0.1$
- **Edge-Weighted:** $\alpha_1 = 0.3, \alpha_2 = 0.1, \alpha_3 = 0.2, \alpha_4 = 0.1, \alpha_5 = 0.3$

Figure 7: First frame of result using Edge-Weighted α values

Figure 8: First frame of result using Temporal-weighted α values

Figure 9: First frame of result using Uniform-weighted α values

Across these configurations, we observed minimal differences in the seam carving results. For instance, the optimal 3D minimum cut exhibits little variation across the manifold, demonstrating the stability of the graph-cut algorithm regardless of α tuning. However, our method shows slight limitations in preserving temporal coherence, especially in foreground-heavy videos, where seam removals can vary significantly between frames. While temporal coherence is well-preserved in the airplane video, some variations in seams are observed in the rat video.

(a) 3D min-cut (frame 1) (b) Independent cut (frame 1) (c) Our method (frame 1)

(d) 3D min-cut (frame 40) (e) Independent cut (frame 40) (f) Our method (frame 40)

(g) 3D min-cut (frame 80) (h) Independent cut (frame 80) (i) Our method (frame 80)

Figure 10: Comparison of our implemented 3D Graph-cut, independent seam removal and our method

(a) Original (b) Rubinstein (c) Our method

Figure 11: Comparison between Rubinstein and our method (video taken from [2])

We compare our results in certain scenarios with [2] as shown in figure 10 and 11.

Web Demo

We provide a user-friendly web demo using Streamlit, enabling easy access to both image and video retargeting functionalities. Users can upload an image or video, specify desired dimensions, and adjust parameters interactively. The demo processes the input using the seam carving algorithm and displays the retargeted output in real-time, making it convenient for experimentation and visualization.

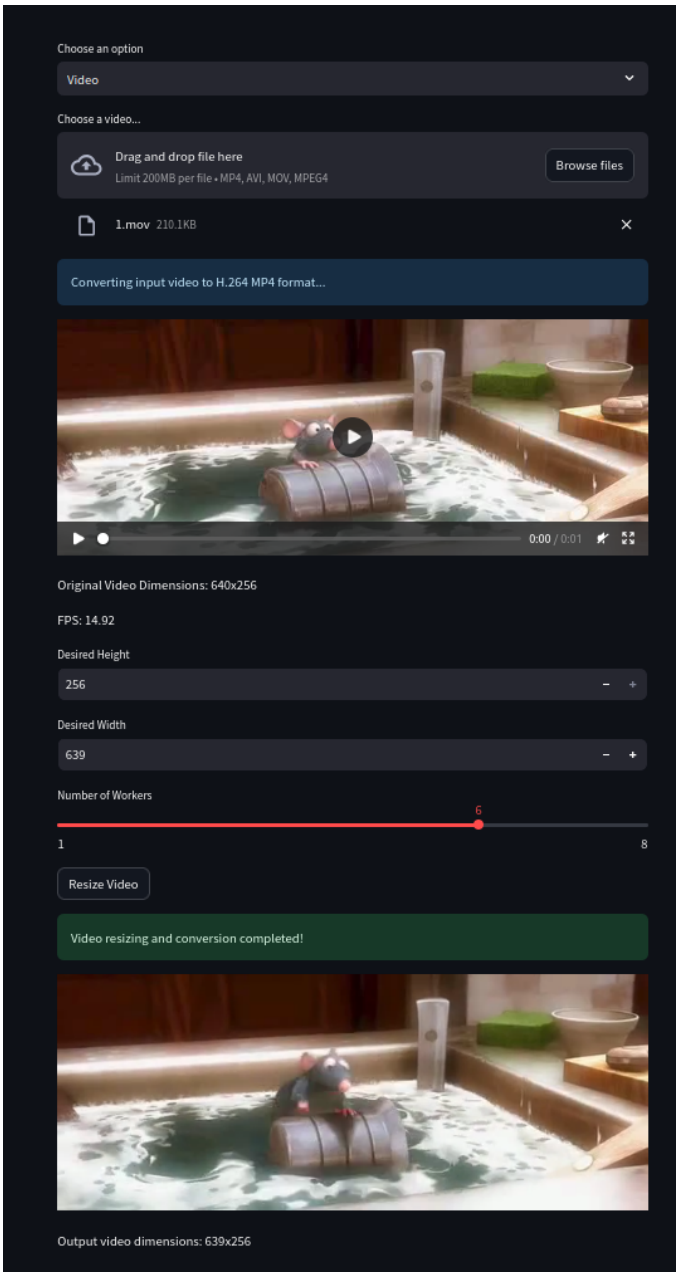

Figure 12: User Interface of Our Web Demo

Conclusion and Future Work

In this study, we successfully implemented the original seam carving algorithms for image and video retargeting. Additionally, we used an enhanced method incorporating look-ahead energy for video seam carving.

In video retargeting, our graph-cut-based algorithm achieved satisfactory results on certain test samples while significantly reducing computation time and consistently matched the quality achieved by Rubinstein's algorithm.

References

[1] S. Avidan and A. Shamir, "Seam carving for content-aware image resizing," in *ACM SIGGRAPH 2007 Papers*, SIGGRAPH '07, (New York, NY, USA), p. 10–es, Association for Computing Machinery, 2007.

[2] M. Rubinstein, A. Shamir, and S. Avidan, "Improved seam carving for video retargeting," *ACM Trans. Graph.*, vol. 27, p. 1–9, Aug. 2008.

[3] A. Shamir and S. Avidan, "Seam carving for media retargeting," *Commun. ACM*, vol. 52, p. 77–85, jan 2009.

[4] V. Setlur, S. Takagi, R. Raskar, M. Gleicher, and B. Gooch, "Automatic image retargeting," in *Proceedings of the 4th International Conference on Mobile and Ubiquitous Multimedia*, MUM '05, (New York, NY, USA), p. 59–68, Association for Computing Machinery, 2005.

[5] R. Gal, O. Sorkine, and D. Cohen-Or, "Feature-aware texturing," in *Proceedings of the 17th Eurographics Conference on Rendering Techniques*, EGSR '06, (Goslar, DEU), p. 297–303, Eurographics Association, 2006.

[6] M. Grundmann, V. Kwatra, M. Han, and I. Essa, "Discontinuous seam-carving for video retargeting," in *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 569–576, 2010.