Proceedings of Parallel Scientific Computing
ID5130 Jan-May 2024
May 3, 2024, IIT Madras, India

HASH2024/PRAGMA-32121

ON THE PARALLELIZATION OF SEAM CARVING

Archish S *
ME20B032
Department of Mechanical Engineering
Indian Institute of Technology Madras
Email: archish@smail.iitm.ac.in

Nilesh Balu *
ME20B121
Department of Mechanical Engineering
Indian Institute of Technology Madras
Email: me20b121@smail.iitm.ac.in

ABSTRACT

Seam Carving is an image operator that dynamically adjusts image dimensions by considering the contents of the image. The algorithm involves the calculation of the gradient-based energy of the image, followed by the removal of 8-connected paths of low-energy pixels, called seams. At the end of this operation, the key features of the image are still preserved. Seam carving operates linearly in the number of pixels, making resizing linear in the number of seams to be removed or inserted. Real-time computation of seams is time-consuming. To address this, we propose a parallel implementation of the algorithm, focusing on phases that involve independent computations for each pixel and seam, viz., the energy calculation and seam identification phases. To this end, we demonstrate the parallelism using OpenMP, MPI, and OpenACC frameworks and compare their performance. Additionally, we utilize this algorithm as an object removal tool that removes a set of pixels selected by the user, ensuring minimal distortion of the image.

Key Words: Image resizing, Content-aware image manipulation, Convolution, Image seams

INTRODUCTION

Traditional methods of scaling and cropping images often do not take into consideration the contents of the image. As a result, these techniques may lead to undesirable distortions or loss of crucial information. Content-aware image manipulation offers a way to go about this while preserving important content and minimizing distortion. We utilize *Seam carving*, an image operator that analyses the contents of the image and identifies *seams* or paths of the least energy, that can then be removed to resize the image. A *seam* is an 8-connected path of pixels that spans the image either from top to bottom or left to right. By successively removing (or inserting) seams, the size of an image can be reduced (or increased) both vertically and horizontally. Furthermore, seam carving can also be extended as an object-removal technique wherein the seams that pass through an object specified by a region of pixels are recursively removed.

Seam carving in real-time is a challenging task as it is linear in the number of pixels, and resizing is, therefore, linear in the number of seams to be removed or inserted. To address this bottleneck, we propose parallel implementations of seam carving.

Related Work

Avidan and Shamir in [1] proposed a way to resize images by taking into consideration the content of the image. They achieved this by calculating the gradient of the image and identifying 8-connected paths that run either vertically or horizontally. This result was much "smarter" than the traditional methods of resizing, viz., cropping or scaling, in that the important information conveyed in the image is retained with minimal distortions. Since then, multiple works have dealt with optimizing the performance of this operation. These have been achieved both by proposing modified algorithms and parallelizing the already existing ones.

Lin et al. in [2] proposed a better way to calculate the energy

*The authors contributed equally to this work.

of an image by using an *accumulative energy* based energy function. This method further reduces the distortions of the images by ensuring that the seams are not concentrated at one particular region of the image. Han et al. in [3] proposed a "modified energy function based on *wavelet decomposition*. This method effectively reduces the distortion of the image along the edges of important objects. In this manner, the performance of the operation in terms of the quality of the result is improved.

Duarte and Sendag in [4] published their attempt to optimize the seam carving algorithm using CUDA-enabled GPUs. They discuss modified algorithms that enable them to achieve good memory coalescence and better spatial locality. In this manner, the performance in terms of the computation time and speed-up is improved.

Applications

The primary application of seam carving is in fields like web design, where images play a key role in conveying information. With the prevalence of mobile devices, it's crucial that images convey information effectively across various screen sizes. Seam carving's "non-destructive" nature makes it a powerful compression tool. This is especially useful in fields like medical imaging, as this ensures that critical information remains intact. The research community is still finding applications where the advantages of seam carving can be exploited. Daldali and Souhar in [5], for instance, have exploited a parallel seam carving algorithm to improve the efficiency of optical character recognition systems and handwritten recognition systems.

Present Study

In this project, we present two implementations of the seam carving operation: one using C and another using C++. To meet the demands of real-time processing, we parallelize the algorithm using the OpenMP, MPI, and OpenACC frameworks to enhance its performance and enable seamless integration into applications requiring dynamic image resizing with minimal latency. The performance boost obtained on all three frameworks is compared, and inferences are drawn. Further, we test the operator's usage as an object-removal tool by removing seams that pass through a user-specified set of pixels.

SEAM CARVING

Avidan and Shamir in [1] propose to remove pixels that are unnoticeable and blend with the surroundings. This leads to an energy-based approach where the optimal strategy is to preserve the energy, thereby removing pixels with the lowest energy. To prevent the image from breaking, we use the strategy of seam carving.

Formally, let $\boldsymbol{I}$ denote an $n \times m$ image, we define the vertical $\boldsymbol{s}^x$ and the horizontal $\boldsymbol{s}^y$ seams as piece-wise 8-connected set of pixels

$$\boldsymbol{s}^x = \{s_i^x\}_{i=1}^n = \{(x(i), i)\} : \forall\, i\, |x(i) - x(i-1)| \leq 1$$
$$\boldsymbol{s}^y = \{s_j^y\}_{j=1}^m = \{(j, y(j))\} : \forall\, i\, |y(j) - y(j-1)| \leq 1$$

where $x : [1, \ldots, n] \mapsto [1, \ldots, m]$ and $y : [1, \ldots, m] \mapsto [1, \ldots, n]$ are maps along the horizontal and vertical axes. The pixels along the path of the seam $\boldsymbol{s}$ are simply $\boldsymbol{I}_{\boldsymbol{s}} = \boldsymbol{I}(\boldsymbol{s})$. Removal of a vertical seam $\boldsymbol{s}^x$ (horizontal seam $\boldsymbol{s}^y$) from an $n \times m$ image yields locally shifted $(n-1) \times m$ image ($n \times (m-1)$ image) with minimal visual distortion.

Minimum Seam

As established previously, by removing the seams of the lowest energy or the minimum seam, the resulting image experiences minimal breaking while re-sizing. We employ an energy-based approach to identify the minimum seam as

$$\boldsymbol{s}^* = \arg\min_{\boldsymbol{s}} E(\boldsymbol{s})$$

where E denotes the energy function that is computed as

$$E(\boldsymbol{s}) = \sum_i e(\boldsymbol{I}(s_i)).$$

e is the energy operator that may be defined as

$$e_1(\boldsymbol{I}) = \left|\frac{\partial \boldsymbol{I}}{\partial x}\right| + \left|\frac{\partial \boldsymbol{I}}{\partial y}\right| \quad \text{or}$$
$$e_2(\boldsymbol{I}) = \sqrt{\left(\frac{\partial \boldsymbol{I}}{\partial x}\right)^2 + \left(\frac{\partial \boldsymbol{I}}{\partial y}\right)^2}.$$

In this project, we utilize the e_2 energy operator applied to a gray-scale image. Further, we apply a Gaussian kernel on the gray-scale image to reduce the effects of inherent noise.

Computing the Energy

In order to compute the derivative of images, we approximate the derivative using the finite difference technique as

$$\frac{\partial f(x,y)}{\partial x} = \frac{f(x+1,y) - f(x,y)}{1},$$

which essentially translates to convolving the kernel $\begin{bmatrix}1 & -1\end{bmatrix}$ on the image. The convolution of a kernel K of dimensions $k \times k$ on an $n \times m$ image $\boldsymbol{I}$ is defined as

$$e(\boldsymbol{I}(x,y)) = \sum_{i=-\lfloor k/2 \rfloor}^{\lfloor k/2 \rfloor} \sum_{j=-\lfloor k/2 \rfloor}^{\lfloor k/2 \rfloor} \boldsymbol{I}(x+i, y+j) K(\lfloor k/2 \rfloor + i, \lfloor k/2 \rfloor + j).$$

By using a Sobel Kernel

$$G_x = \begin{bmatrix}+1\\+2\\+1\end{bmatrix} \times \begin{bmatrix}+1 & 0 & -1\end{bmatrix}$$

$$G_y = \begin{bmatrix}+1\\0\\-1\end{bmatrix} \times \begin{bmatrix}+1 & +2 & +1\end{bmatrix}$$

the derivative can be approximated to a higher-order accuracy. The energy for the Image 1a is depicted in Fig. 1b.

Computing the Minimum Seam

To compute the optimal vertical seam, we use a dynamic programming paradigm where $M(i\,j)$ denotes the minimum cumulative energy for all possible seams that pass through (i, j), which can be recursively computed as

$$M(i, j) = e(i, j) + \min\left(M(i-1, j), M(i-1, j), M(i-1, j+1)\right).$$

At the end of the computation, the index j along $i = n$ would represent the minimal connected vertical seam when traced back. A similar approach can be followed to compute the optimal horizontal seam. The cumulative energy for the Image 1a computed along the vertical and the horizontal directions are depicted in Fig. 1d, and the corresponding minimum seams are shown in Fig. 1c.

IMPLEMENTATION

We implement Algorithm 1 in C using 1D arrays and in C++ using vectors, and each variant is parallelized using OpenMP, MPI, and the OpenACC frameworks. The avenues of parallelism for each framework are briefly discussed in this section.

Parallelization Strategy

OpenMP OpenMP enables quick parallelization of the convolutions. However, the calculation of the cumulative energy involves loop-carried dependency and, hence, cannot be doubly parallelized. The unrolling used to compute the pixels that are to be removed, and the removal is also parallelized.

Algorithm 1: Seam Carving

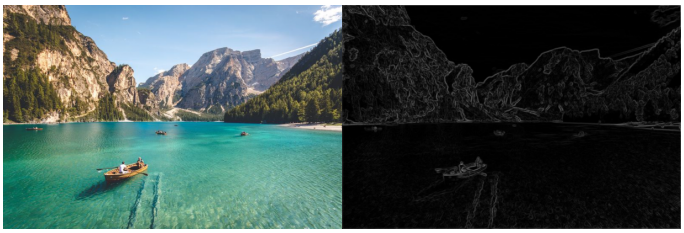
Input: image $\boldsymbol{I}$, image dimensions n, m, scale η
Initialize: $t = 0$, energy $E[N-1][M-1]$, min_energy $M[N][M]$, seam $S[N][M]$, remove_seam $R[N]$

convert $\boldsymbol{I}$ to gray scale ($\boldsymbol{g}$)
blur $\boldsymbol{g}$ using $\mathcal{N}$ kernel
compute $E_x[N-1][M-1]$ and $E_y[N-1][M-1]$ as convolved output of $\boldsymbol{g}$ with kernel G_x and G_y
for $t \leq m - \eta m$ **do**
  compute $E[i][j]$ as $E[i][j] = \sqrt{E_x[i][j]^2 + E_y[i][j]^2}\ \forall i, j \in [N-1] \times [M-1]$
  for $i = 1;\ i < N;\ i{+}{+}$ **do**
    for $j = 1;\ j < M-1;\ j{+}{+}$ **do**
      $S[i][j] = \underset{q=\{j-1,\,j,\,j+1\}}{\arg\min}\ M[i-1][q]$
      $M[i][j] = E[i][j] + E[i-1][S[i][j]]$
  compute $R[N-1] = \underset{j \in [m]}{\arg\min}\ M[N-1][j]$ and unroll S to obtain $R[i] = S[i][R[i+1]]\ \forall\, i \in [2, N-1]$ set $R[1] \leftarrow R[2]$ & $R[N] \leftarrow R[N-1]$
  remove pixel $R[i]$ from i^{th} row of $\boldsymbol{I}$
  for $i = 1;\ i \leq N;\ i{+}{+}$ **do**
    start $= \min_{p=\{i-1,\,i,\,i+1\}} R[i] - 1$ and
    end $= \max_{p=\{i-1,\,i,\,i+1\}} R[i] + 1$
    re-compute $E_x[\text{start} : \text{end}][M-1]$ and $E_y[\text{start} : \text{end}][M-1]$

Output: re-sized image $\boldsymbol{I}'$ with dimensions $n \times \eta m$

MPI We use a row-block decomposition strategy to scatter the image to multiple processors. Each thread computes the convolution simultaneously (with necessary halo points). Since the calculation of cumulative energy depends on the entire row space, it cannot be decomposed about the rows. However, we can adopt a column-block decomposition strategy, but we realize that the communication overhead in broadcasting the energy values after each row is more than the gain due to parallelization. Hence, we resort to computing the cumulative energies redundantly on all processors. The unrolling and the removal are also parallelized.

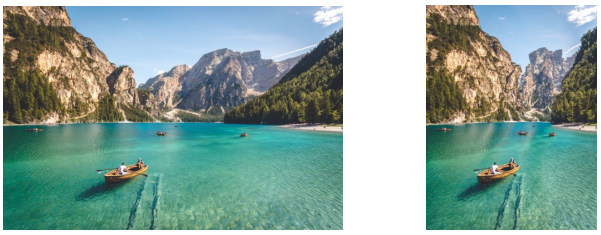
OpenACC We use a parallelism scheme similar to OpenMP, where we compute the convolutions on the accelerator. Parallelising the calculation of cumulative energy resulted in misaligned images.



(a) Source Image (b) Computed Energy

(c) Identified Minimum Seams (d) Cumulative Energy

FIGURE 1: SEAM CARVING: INTERMEDIATE STEPS



(a) Source Image (b) 50% Carved Vertical

(c) 50% Carved Horizontal

FIGURE 2: SEAM CARVING AS AN IMAGE RE-SIZING TOOL

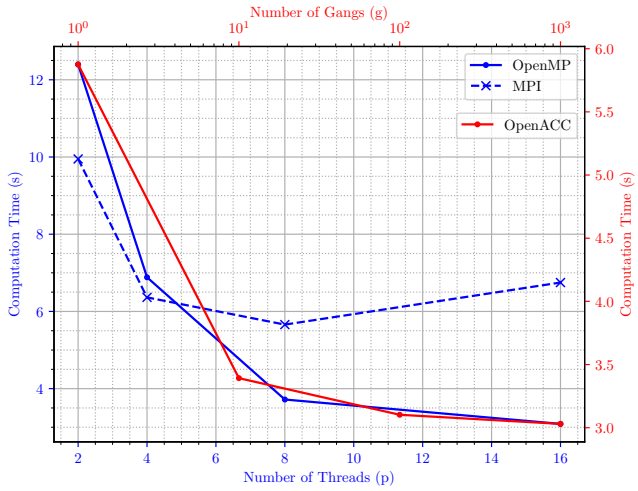


FIGURE 3: EXECUTION TIME FOR AN 800 × 528 IMAGE.

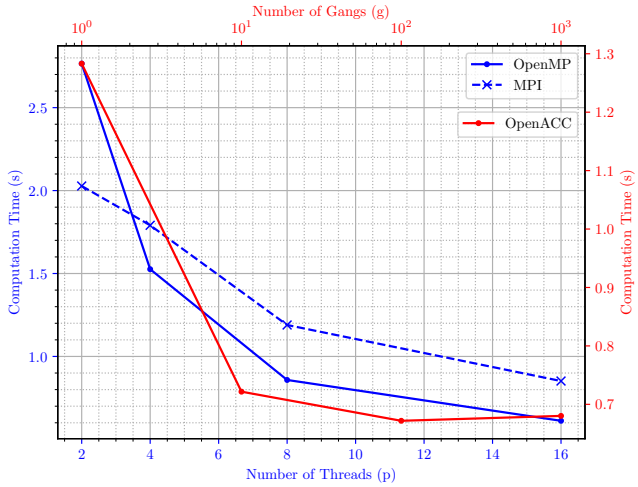


FIGURE 4: EXECUTION TIME FOR AN 400 × 264 IMAGE.

RESULTS

Image Re-sizing

We have employed this technique as an image re-sizing tool. To re-size a given image $\boldsymbol{I}$ from $n \times m$ to $n \times \eta m$ (for $0 < \eta < 1$), we successively remove $c = m - \eta m$ seams from $\boldsymbol{I}$. The results are shown in Fig. 2.

Execution Time

The results of the C++ implementation are shown here. The seam carving operation is performed to remove 200 seams on an image of size 800 × 528, labeled 'Image 1'. The serial code takes 20.553 seconds to compute. The execution times for the different parallel implementations are plotted against the number of threads or gangs in Fig. 3 for the different parallel paradigms implemented. The same is done for a smaller image of the same content of size 400 × 264. labelled 'Image 2'. The serial code takes 4.4698 seconds to run. The execution times for the parallel implementations are plotted in Fig. 4.

From Fig. 3 and Fig. 4, it is evident that the time taken for the seam carving operation is higher in the case of the larger image, indicating that seam carving is indeed linear in the number of pixels. It is also observed that OpenACC gives the most speed-up among the three implementations, and MPI gives the least speed-up. This observation is in accordance with our parallelization strategy.

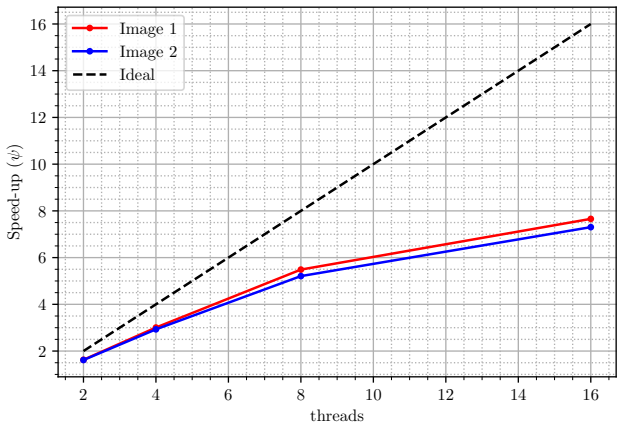


FIGURE 5: SPEED-UP FOR OPEN-MP.

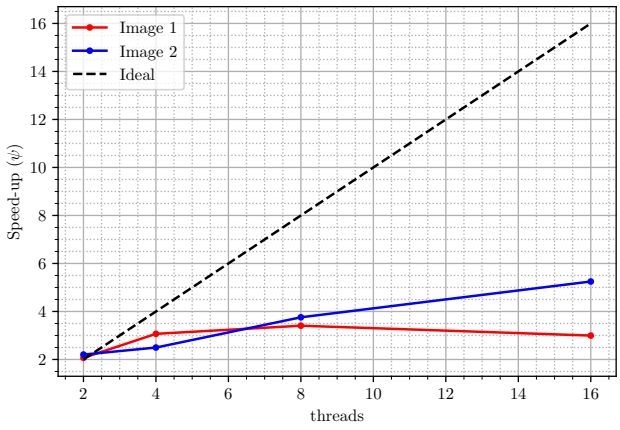


FIGURE 6: SPEED-UP FOR MPI.

Speed-up

The speed-up is defined as

$$\psi(N,p) = \frac{T_s}{T_p}$$

where N is the problem size, p is the number of threads, T_s is the execution time of the serial code, and T_p is the execution time of the parallel program. The speed-up is calculated for both the image sizes and is compared in Fig. 5, Fig. 6 and Fig. 7.

Object Removal

As mentioned earlier, we have employed this operator as an object-removal tool. In our implementation, the user specifies a particular rectangle by clicking on the top-left and bottom-right corners of the rectangle on the image. This rectangular region will be removed by the code. A mask is created, wherein the pixels of the image lying in this region are given a large magnitude of negative energy, which leads to the algorithm removing the rectangular region first. The algorithm also ensures that the seam is the minimum energy seam, thereby minimizing distortions around the object to be removed. However, this algorithm has its own limitations in object removal, as mentioned in [1]. The results are shown in Fig. 8.

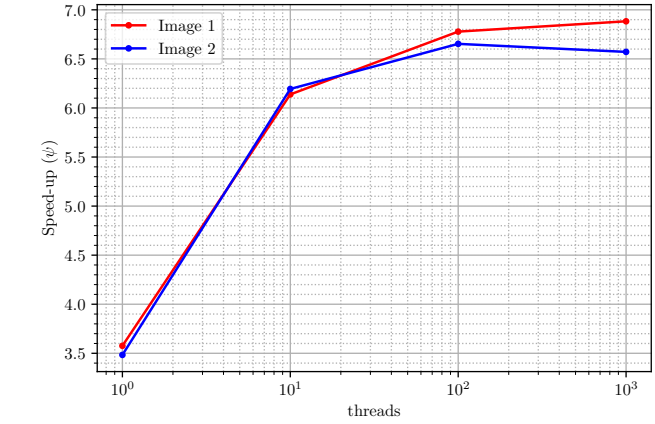


FIGURE 7: SPEED-UP FOR OPEN-ACC.

(a) Before Object Removal (b) After Object Removal

FIGURE 8: SEAM CARVING AS AN OBJECT REMOVAL TOOL

CONCLUSION

We presented a parallel implementation of content-aware resizing of images using *seam carving*. Seams are computed as 8-connected minimal energy paths that are removed successively to re-size an image. In addition, we also present a modification that allows object removal, where seams that pass through user-specified pixels are removed. We also demonstrate speed-up in OpenMP, MPI, and OpenACC paradigms.

ACKNOWLEDGMENT

We thank Prof. Kameswararao Anupindi for offering this course and for his unwavering support and guidance throughout the semester, which has played a pivotal role in the success of our project. We also thank Prof. Manish Anand for providing access to a 16-core workstation with an Nvidia graphics card, a valuable resource for our work.

REFERENCES

[1] Avidan, S., and Shamir, A., 2007. "Seam Carving for Content-Aware Image Resizing". *SIGGRAPH,* ***26***, July.

[2] Lin, Y., Niu, Y., Lin, J., and Zhang, H., 2016. "Accumulative Energy-Based Seam Carving for Image Resizing". In 2016 17th International Conference on Parallel and Distributed Computing, Applications and Technologies (PDCAT), pp. 366–371.

[3] Jong-Woo Han, Kang-Sun Choi, Tae-Shick Wang, Sung-Hyun Cheon, and Sung-Jea Ko, 2009. "Improved seam carving using a modified energy function based on wavelet decomposition". In 2009 IEEE 13th International Symposium on Consumer Electronics, IEEE, pp. 38–41.

[4] Duarte, R., and Sendag, R., 2012. "Accelerating and Characterizing Seam Carving Using a Heterogeneous CPU-GPU System".

[5] Daldali, M., and Souhar, A. "Distributed Multi-Agent Implementation of Text Line Segmentation Using Parallel Seam Carving".

APPENDIX A. EXAMPLES

(a) $n \times m$ (b) $0.8n \times m$ (c) $0.6n \times m$ (d) $0.6n \times 0.6m$

FIGURE 9: IMAGE RE-SIZING: Example 1

(a) $n \times m$ (b) $n \times 0.8m$ (c) $n \times 0.6m$ (d) $0.6n \times 0.6m$

FIGURE 10: IMAGE RE-SIZING: Example 2

(a) $n \times m$ (b) $0.9n \times 0.9m$ (c) $0.8n \times 0.8m$ (d) $0.7n \times 0.7m$

FIGURE 11: IMAGE RE-SIZING: Example 3