COL783: Digital Image Analysis

Assignment 3: Seam Carving

November 23, 2023

Collaborators:
- Shashank Krishna Vempati (2022AIY7509)
- Kaustubh R Borgavi (2023SIY7539)

Introduction

The report consists of two parts:

- Seam Carving
- Learning Based Seam Carving

All seam removals are added as gifs in the attached drive link **

Part 1: Seam Carving

Seam carving, also known as content-aware image resizing, is a technique used in image processing to intelligently resize images while preserving their important content. Unlike traditional resizing methods that uniformly scale images, seam carving considers the image content to retain important features like objects, structures, and details.

Figure 1: Input Image (532x706)

Energy Map using DCT-IDCT:

Energy Map Computation:

Seam carving begins by computing an "energy map" for the image. The energy map represents the importance of each pixel in the image's content. High-energy pixels typically correspond to edges, textures, or other significant features. The energy map is crucial for determining which seams to remove or insert during the resizing process.

DCT-IDCT Compression and Reconstruction:

The combined use of DCT for compression and IDCT for reconstruction is a key aspect of the seam carving process. DCT is a transform that converts spatial information into frequency information, allowing for efficient compression by representing the image in the frequency domain. The inverse process, IDCT, is used to reconstruct the image from the compressed representation.

By applying DCT, high-frequency components in the image are separated from low-frequency components. This separation allows for the removal of less essential information (seams) while retaining crucial image structures. The energy map, computed earlier, guides the seam removal process by identifying seams that contribute less to the overall content. Retaining high-frequency components aids in better edge detection, enhancing the ability to delineate objects within the compressed image. This approach achieves efficient compression while preserving significant image features.

Energy Map Derivation:

After performing the DCT-IDCT process, the energy map is derived by computing the image's derivative along both the x and y axes. This derivative operation highlights changes in intensity or energy across the image, helping to identify important features and structures. This derivative-based approach aligns with the concept of content-aware resizing, ensuring that the resizing operation is performed in a way that considers the content's significance.

Reference to "Image Retargeting in Compressed Domain":

The mentioned paper titled "Image Retargeting in Compressed Domain" likely provides additional details and insights into the specific techniques and optimizations involved in the DCT-IDCT-based seam carving approach.

In summary, the combination of DCT and IDCT in seam carving allows for efficient compression and reconstruction while preserving crucial image structures. The energy map, derived from the DCT-IDCT process, guides the seam carving operation in a content-aware manner, ensuring that important features are retained during image resizing.

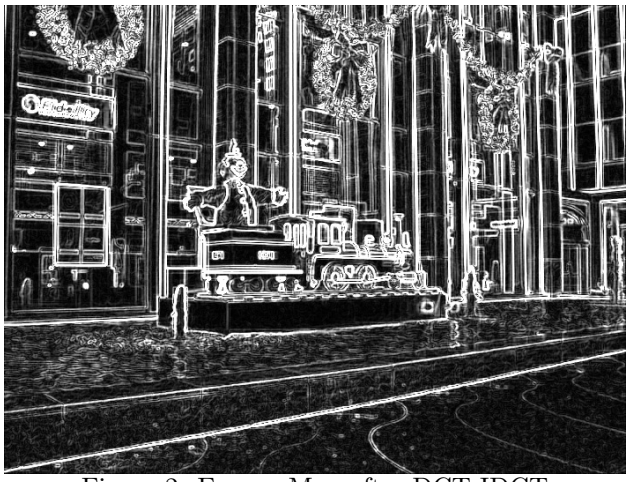


Figure 2: Energy Map after DCT-IDCT

Cumulative Energy Map for Seam Carving is computed after every seam removal/insertion operation.

Energy Map computation equation (Inverse DCT):

$$f(x,y) = \frac{1}{4}C(u)C(v)\sum_{u=0}^{7}\sum_{v=0}^{7}F(u,v)\cos\left(\frac{(2x+1)u\pi}{16}\right)\cos\left(\frac{(2y+1)v\pi}{16}\right)$$

where

$$C(u) = \begin{cases} \frac{1}{\sqrt{8}}, & \text{if } u = 0 \\ \frac{\sqrt{2}}{\sqrt{8}}, & \text{otherwise} \end{cases}$$

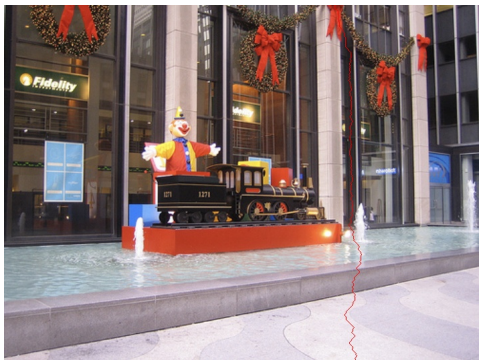
Seam Identification and Removal/Insertion:

Using the energy map, the algorithm identifies seams—connected paths of low-energy pixels that span from the top to the bottom (or left to right) of the image. These seams represent areas of relatively less importance in the image content using dynamic programming as suggested in the paper.

The algorithm removes the identified seams by iteratively carving or deleting these seams from the image. This deletion can be performed vertically or horizontally, depending on the resizing direction.

Seam carving isn't solely about removing seams to reduce the size of an image; it also includes seam insertion to expand or enlarge the image. The identified seams are inserted into the image, typically in areas where expansion is needed. These seams are added by duplicating or interpolating pixels to increase the image's width or height.

The following example demonstrates the insertion of seams to expand the width while deleting seams to reduce the height of an image based on the energy map.



(a) ith Seam Insertion

(b) jth Seam Removal

Figure 3: Seams while processing

Figure 4: Output Image after Seam Carving (480x735)

A Seam Carving GIF is generated and stored in the "$output\_gif$" directory for each image in part 1.

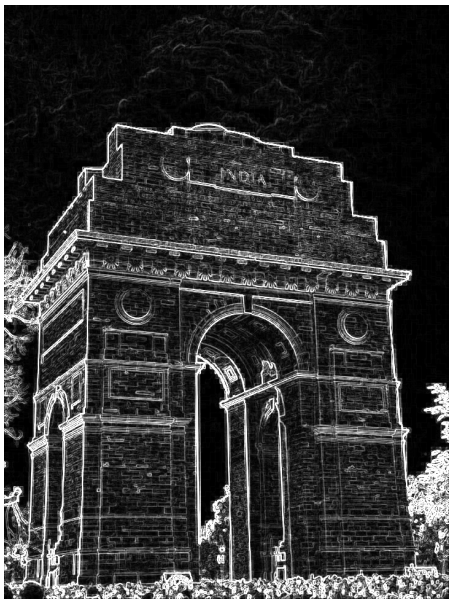
Some failure Cases:

Set: Photo taken by our camera

Seam carving may fail to produce satisfactory results when crucial image features are not accurately identified in the energy map, leading to unintended distortions or loss of meaningful content. Additionally, limitations may arise in scenarios where preserving the global structure of the image is essential, such as in complex compositions or highly detailed scenes.

In the following example, the image contains abundant details, resulting in the distortion of its primary structure.

Figure 5: Input Image



(a) Energy Map (DCT-IDCT)

(b) Output Image (10% seams removed on both sides)

Figure 6: Seams while processing

Analysis of why DCT-IDCT produces a better energy map:

Energy Compaction Property:

The DCT has the property of energy compaction, meaning that most of the signal energy is concentrated in a small number of coefficients. In image compression, this property allows for more efficient representation of the image with fewer coefficients, reducing the redundancy in the data. When performing the IDCT to reconstruct the image, the dominant coefficients contribute more to the overall appearance, resulting in a more accurate representation of the image details.

Orthogonality and Decorrelation:

DCT basis functions are orthogonal to each other, and they can be less correlated with the image data compared to spatial operators like Sobel or Scharr filters. This orthogonality and reduced correlation can lead to a more decorrelated set of coefficients, which is beneficial for various image processing tasks. In contrast, spatial operators may respond strongly to certain patterns or orientations, potentially leading to less effective energy maps.

Global Information Handling:

DCT is applied globally to the entire image or image blocks, considering the entire context rather than local neighborhoods. This global perspective helps in capturing the overall structure and energy distribution in the image, making the DCT more effective in representing image features consistently.

Part 2: Learning Based Seam Carving

In Part 2, we examined a research paper titled "Wide-Context Semantic Image Extrapolation," presented at CVPR 2019 [1], with the goal of adapting the network for the purpose of performing seam carving.

Model architecture and Reproducing the results of the paper

The paper aims to solve problem of image extrapolation which is a task which falls under the umbrella of image-to-image translation. The learns the mapping between the input images and required output image and generates the missing regions of the image by utilizing the existing context such that the output image produced is coherent as whole. The two major issues here are size expansion and one-side constraints. They propose a semantic regeneration network with several special contributions and use multiple spatial related losses to address these issues. The model architecture is as follows:

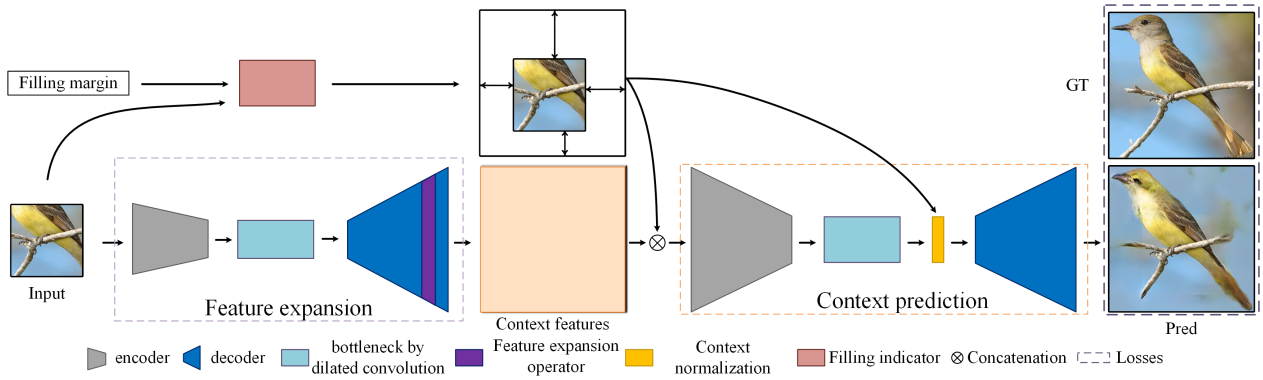


Figure 7: Outpaint SRN Architecture

Given a cropped image with partial image, the model learns to complete the image using the existing context. The models consists of two stages. The first consists of a Feature expansion network [FEN] and the second stage is called the Context Prediction Network [CPN]. These two stages are combined to make the generator and the model is trained in an adversarial setup. The discriminator is the one used WGAN-GP architecture as proposed in [2] which shows that the use of weight clipping in WGAN to enforce a Lipschitz constraint on the critic, can lead to undesired behavior and unstable training of the GAN. Hence, the authors use an alternative to clipping weights by penalizing the norm of gradient of the critic with respect to its input which ensure stable training of the generator and the discriminator. The authors training the architecture on the CelebA, Parisview and Cityscapes datasets and have provided their code and trained weights to reproduce their results.

Figure 8: Input image and output from the model on the Cityscapes dataset (left to right)

Figure 9: Input image and output from the model on the Cityscapes dataset (left to right)

Figure 10: Input images and outputs from the model on the Parisview dataset (left to right)

Training the model for our task

Since the model naturally addresses outpainting with a small image patch and a white mask surrounding it, we endeavored to modify the model to process the original input along with an empty white mask to generate retargeted images using the seam-carved GTs produced using the DCT energy functions from Part A of this assignment. As suggested by the authors, we first pretrain the generator without the discriminator with only the $L1$ reconstruction loss and then finetune the model using the discriminator along with the adversarial loss.

Dataset and data preprocessing

We use the dataset of 300 images provided to us for this assignment to train the model. The dataset contains images featuring4 various outdoor object/locations. As a preprocessing step, we use the Seam-carving algorithm [3] from Part 1 to generate the corresponding seam-carved ground truths for each image in the dataset. Some paired samples from the dataset are as follows:

(a) Input images (b) Output images (Seam-carved)

Figure 13: Paired samples from the dataset used for training the model

We resize the input image to $128 \times 128 \times 3$ and its output to $256 \times 256 \times 3$ before providing them to the model to match the configuration of the model architecture.

Training Approaches Employed:

- To tailor the model for the seam-carving task, an initial training phase involved training for a few epochs, ensuring consistency between the training and test sets. This phase aimed to teach the model to replicate the input image onto a white mask during the decoder stage.
- Subsequently, the model underwent training using our dataset of input-output pairs generated in part-1 for 3200 steps.
- We experimented with training the generator and discriminator for an initial set of steps, followed by exclusive training of the generator for additional steps, leading to improved outcomes in the process.
- To tailor the model for the outpainting task, an initial training phase involved training for a few epochs, ensuring consistency between the training and test sets. This phase aimed to teach the model to replicate the input image onto a white mask during the decoder stage. Subsequently, the model underwent training using our dataset of input-output pairs generated in part-1.

Loss functions

The $L1$ reconstruction loss used is as follows:

$$L_{recon} = |Y - G(X)| \tag{1}$$

where Y is the GT image and $G(X)$ is the generated image. The WGAN-GP adversarial loss used is defined as follows:

- Wasserstein Distance (or W-distance): The WGAN framework aims to minimize the Wasserstein distance between the distributions of real and generated samples. This distance is calculated as the difference between the expected values of the critic (discriminator) scores for real and generated samples. The formula for the Wasserstein distance is:

  $$\text{Wasserstein Distance} = \mathbb{E}_{\mathbf{x}\sim P_{\text{data}}}[D(\mathbf{x})] - \mathbb{E}_{\mathbf{z}\sim P_{\text{gen}}}[D(G(\mathbf{z}))] \tag{2}$$

  Here, $D(x)$ represents the output of the critic for real samples, $D(G(z))$ represents the output of the critic for generated samples, and E denotes the expectation over the respective distributions.

- Gradient Penalty: The gradient penalty is introduced to enforce a Lipschitz constraint on the critic function (discriminator). It penalizes the norm of the gradient of the critic with respect to interpolated samples between real and generated samples. This penalty encourages the gradients of the critic to be close to 1 (which helps in stable training) and prevents them from becoming too large. The formulation for the gradient penalty in the WGAN-GP is:

  $$\text{Gradient Penalty} = \lambda \cdot \mathbb{E}_{\hat{\mathbf{x}}}[(||\nabla_{\hat{\mathbf{x}}} D(\hat{\mathbf{x}})||_2 - 1)^2] \tag{3}$$

  Here, $\hat{\mathbf{x}}$ represents interpolated samples between real and generated samples. λ is a penalty coefficient. $D(\hat{\mathbf{x}})$ is the output of the critic for the interpolated samples, and $|||\dot{|}||$ denotes the L2 norm.

- The total WGAN-GP adversarial loss is the combination of the Wasserstein distance and the gradient penalty:

  $$L_{adv} = -\text{Wasserstein Distance} + \text{Gradient Penalty} \tag{4}$$

The final loss objective is :

$$L_{total} = \lambda_{recon} \times L_{recon} + \lambda_{adv} \times L_{adv} \tag{5}$$

where λ_{recon} and λ_{adv} are constants for scaling which are taken as mentioned in the paper.

Implementation details

To better stabilize the adversarial training, the model is pretrained first with only reconstruction loss ($\lambda_{\text{recon}} = 5$). Afterwards, we let $\lambda_{\text{adv}} = 0.001$ for fine-tuning SRN until convergence. During training, Adam solver with learning rate 1×10^{-4} is adopted, where $\beta_1 = 0.5$ and $\beta_2 = 0.9$. We train it for a total of 4000 iterations. The graph depicting the change in the total loss across different epochs is as follows:

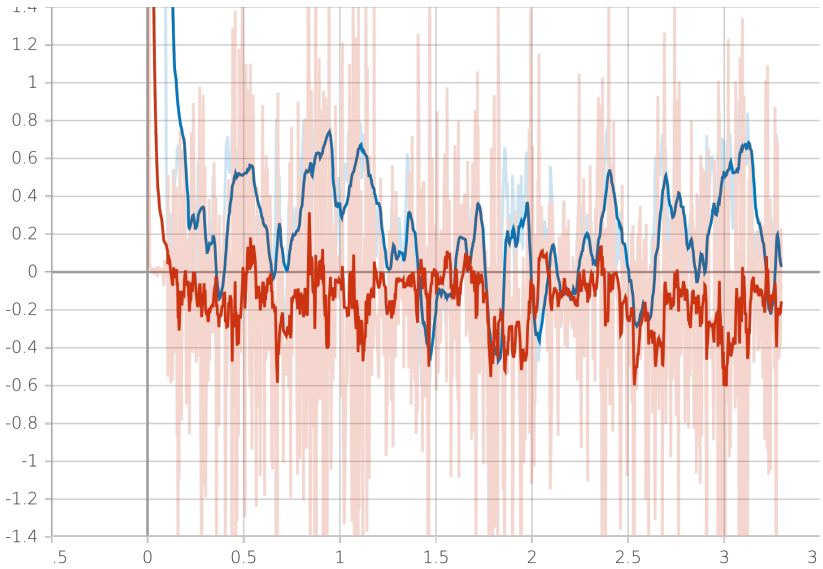


Figure 14: Change in generator and discriminator loss for every 10 iterations

Training batch size is set as 16. The input and output are linearly scaled within the range $[-1, 1]$.

Results

The quantitative results are measured using the SSIM, RMSE and NMI metrics.

The average value across the five sets specified in the ablations.

Using RMSProp optimizer:

SSIM	RMSE	NMI
0.58	10.54	0.26

Table 1: Score on different metrics

Using Adam optimizer:

SSIM	RMSE	NMI
0.37	10.08	0.13

Table 2: Score on different metrics

RMS (Root Mean Square): Lower values indicate better pixel-level similarity between images.

SSIM (Structural Similarity Index): Higher values signify greater structural similarity and visual quality between images.

NMI (Normalized Mutual Information): Higher values suggest stronger mutual information and increased similarity between images.

Key Findings:

Although the model learned to reproduce the input image, the color transfer from the original image did not succeed, as evident in the output results.

It's noteworthy that due to its computational intensity, the selected GAN model limited our training to 3500 steps on GPU instead of the recommended 40000 steps in the paper.

All weights for the tasks mentioned above are uploaded to the google drive.

Drive Link for all the Files

Google Drive

Ablation Study

Part 1: Seam Carving

Set 1: Seam Removal

(a) Input Image (384x700)

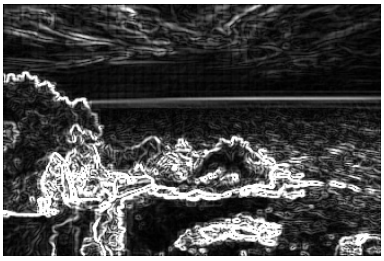
(b) Energy Map

(c) Output Image (300x500)

Figure 15: Seam Carving: Set 1

Set 2: Seam Insertion

(a) Input Image (230x345)



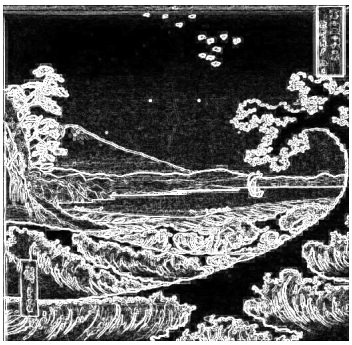
(b) Energy Map

(c) Output Image (300x400)

Figure 16: Seam Carving: Set 2

Set 3: Seam Insertion and Removal

(a) Input Image (502x518)



(b) Energy Map

(c) Output Image (452x558)

Figure 17: Seam Carving: Set 3

Set 4: Seam Insertion

(a) Input Image (171x206)

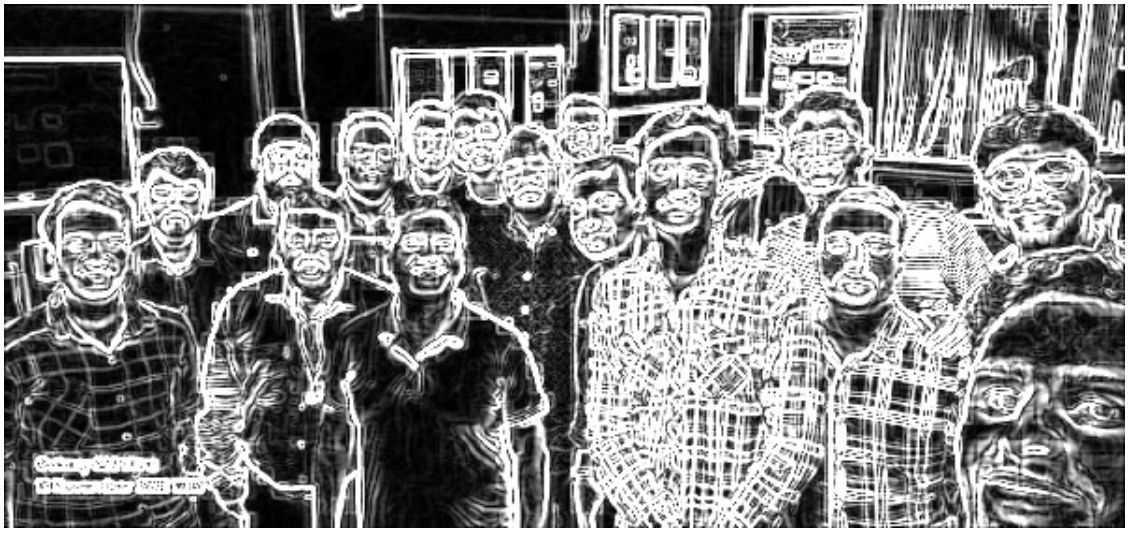
(b) Energy Map

(c) Output Image (171x310)

Figure 18: Seam Carving: Set 4

Set 5: Seam Removal on phone clicked image

(a) Input Image (280x600)

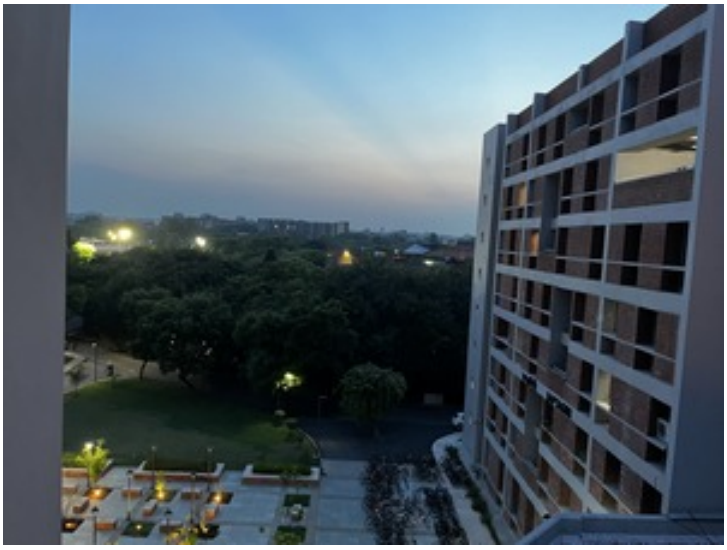


(b) Energy Map

(c) Output Image with 10% seams removed(250x540)

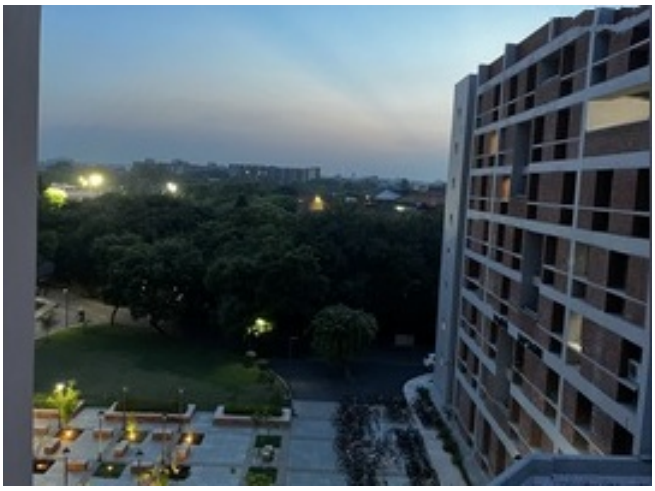
Figure 19: Seam Carving: Set 5

Set 6: Seam Removal on phone clicked image



(a) Input Image (225x300)

(b) Energy Map



(c) Output Image with 10% seams removed(200x270)

Figure 20: Seam Carving: Set 6

Set 6: Seam Insertion (Collage View of Seams)

(a) Input Image (566x370)

(b) Collage of Seam Carving (Sampled at various intervals)

Figure 21: Seam Carving Input and Collage of Seam Carving: Set 6

Figure 22: Output Image (566x450)

Set 7: Seam Carving using different Energy Maps

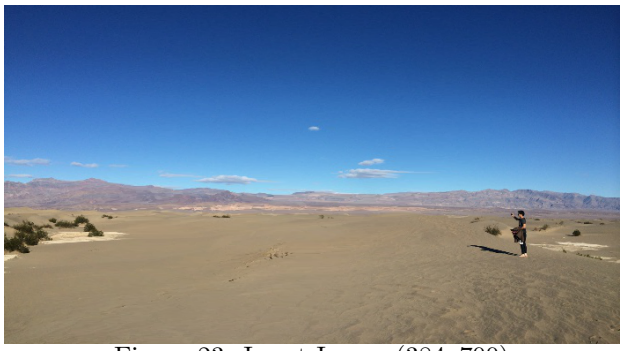
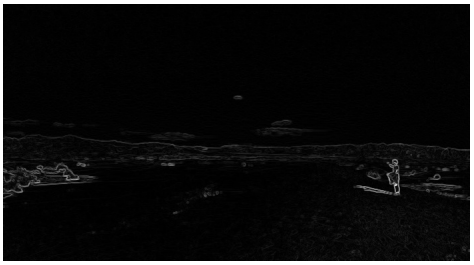


Figure 23: Input Image (384x700)



(a) Energy Map (Simple Sobel Filter based Edge Map)

(b) Output Image (210x475)

Figure 24: Seam Carving : Set 7 (Sobel filter based Energy Map)

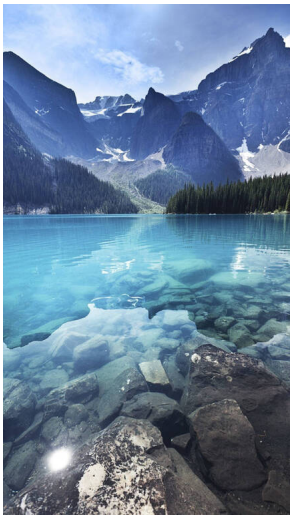
(a) Energy Map (DCT based Energy Map)

(b) Output Image (210x475)

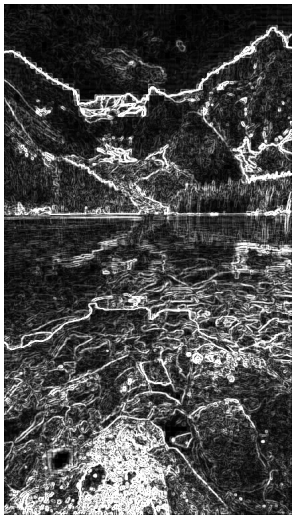
Figure 25: Seam Carving: Set 7 (DCT based Energy Map)

Part 1 on our images

Set 8: Seam Insertion and Removal



(a) Input Image (566x370)



(b) Energy Map

(c) Output Image (452x558)

Figure 26: Seam Carving: Set 5

Part 2: Learning-based Seam Carving

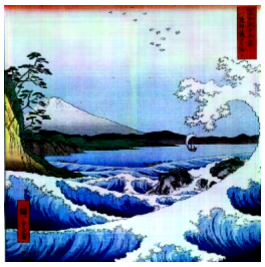
RMSProp as optimizer

(a) Input Image to the model (512x512)

(b) Output Image (384x384)

Figure 27: Set 1)

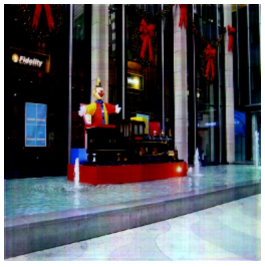
(a) Input Image to the model (512x512)



(b) Output Image (384x384)

Figure 28: Set 2)

(a) Input Image to the model (512x512)



(b) Output Image (384x384)

Figure 29: Set 3)

(a) Input Image to the model (512x512)

(b) Output Image (384x384)

Figure 30: Set 4)

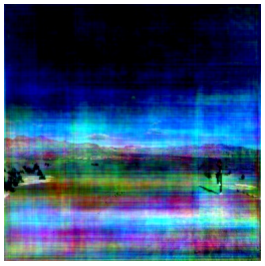
(a) Input Image to the model (512x512)

(b) Output Image (384x384)

Figure 31: Set 5)

Adam as optimizer

(a) Input Image to the model (512x512)



(b) Output Image (384x384)

Figure 32: Set 1)

(a) Input Image to the model (512x512)

(b) Output Image (384x384)

Figure 33: Set 2)

(a) Input Image to the model (512x512)

(b) Output Image (384x384)

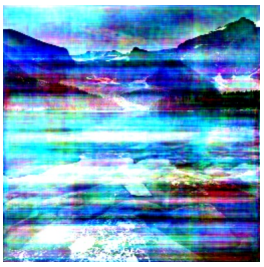
Figure 34: Set 3)

(a) Input Image to the model (512x512)

(b) Output Image (384x384)

Figure 35: Set 4)

(a) Input Image to the model (512x512)



(b) Output Image (384x384)

Figure 36: Set 5)

References

[1] Yi Wang, Xin Tao, Xiaoyong Shen, and Jiaya Jia. Wide-context semantic image extrapolation. In *CVPR*, pages 1399–1408. Computer Vision Foundation / IEEE, 2019.

[2] Ishaan Gulrajani, Faruk Ahmed, Martín Arjovsky, Vincent Dumoulin, and Aaron C. Courville. Improved training of wasserstein gans. In *NIPS*, pages 5767–5777, 2017.

[3] Shai Avidan and Ariel Shamir. Seam carving for content-aware image resizing. *ACM Trans. Graph.*, 26(3):10, 2007.