Project Report

Shreyansh Choudhary

January 13, 2023

Problem Statement

Image retargeting is a technique that adjusts the images into different form factors based on the size of the displaying devices and simultaneously preserve salient features of the image. The three main objectives for Image Retargeting:

- The important content/features of the image should be preserved in the retargeted image.
- The retargeted image should be free from imperfections.
- The salient structures in the original image should not be hampered.

In this project, we need to implement the Forward Energy Seam Carving proposed by "Improved Seam Carving for Video Retargeting" by Rubinstein et al. [5] to reduce the width by 0.75, i.e. to retarget an $m\times$ n image to $\lfloor 0.75m \rceil \times n$.

We need to further need to use the same to retarget video with frame size $m\times$ n to frame size $\lfloor 0.75m \rceil \times \lfloor 0.5n \rceil$

Method

Related Work

Image resizing is a standard tool in many image processing applications. It works by uniformly resizing the image to a target size. Effective Resizing should not only use geometric constraints, it should keep in mind the content of the image. Hence, cropping or scaling is not enough. Both can lead to visual artifacts in the image, making it unnatural or artificial.

Hence, it is crucial to retarget the image to maintain the important features of the image. These features can either be detected by top down methods or bottom up methods. Top down methods [6] use tools such as face detectors to detect important regions in the image, whereas bottom-up methods [2] rely on visual saliency methods to construct a visual saliency map of the image. Once the saliency map is created, we can crop the unimportant parts of the images.

Further, a non linear, data dependent scaling was proposed [3] for image and video retargeting. They find the Region-Of-Interest and construct a fish-eye warp that applies piecewise linear scaling in each dimension of the image, maintaining the ROI and warping the rest of the image.

Shamir et al.[1] proposed seam carving for image retargeting and used dynamic programming to find the optimal seam iteratively. The Use of seams for image editing is prevelant. Seam carving is a method to reduce the dimension of an image in one dimension while preserving important information.

Video retargeting is a much complex process than image retargeting due to added temporal dimension. Temporal coherence is much important and pleasing to watch. Extending the work of Shamir et al.[1], Rubinstein et al. [5] proposed a seam carving approach using graph-cuts to extend the idea to the video retargeting.

Preliminaries

- Seam Carving - A seam is a monotonic and connected path of pixels going from the top of the image to the bottom, or from left to right. reduced by one either in the horizontal or the vertical dimension. Seam carving uses an energy function defined on the pixels and successively removes minimum energy paths from the image. For video, we assume that the camera shot sequence are continuous to apply the methods. We can proceed with the following ways:-

  1. Applying Image retargeting method on each frame of the video independently. But this can cause jittery effects, as seam carving on each frame independently constructs locally optimally seams that can be totally different over time. Hence, this technique is not enough
  2. We can search for image regions which are lower importance in all the frames.This is done by computing the energy on every image independently and then taking the maximum energy value at each pixel location, thus reducing the problem back to image retargeting. This might perform better in which the objects in the image aren't moving much. However, in much complex cases where either camera motion is there or motions in multiple directions are happening, the static seam technique may fail miserably, implying that seams should be allowed to adapt over time.

  $$E_{\text{spatial}}(i,j) = \max_{t=1}^{N}\{|\frac{\partial}{\partial x}I_t(i,j)| + |\frac{\partial}{\partial y}I_t(i,j)|\}$$

  $$E_{\text{temporal}}(i,j) = \max_{t=1}^{N}\{|\frac{\partial}{\partial t}I_t(i,j)|\}$$

  $$E_{\text{global}}(i,j) = \alpha \cdot E_{\text{spatial}} + (1-\alpha) \cdot E_{\text{temporal}}$$

  Due to failure of the above techniques, we define a video seam as a connected 2D manifold "surface" in space-time ($X \times T$) that cuts through the video 3D cube ($X \times Y \times T$). This way we have both the properties that the seam is connected (maintaining temporal coherency) and the seam in each frame is dynamic over time. We cannot extend this idea with dynamic programming and hence, we use graph cut technique.

- Graph Cut - The formulation of the graph from the image as follows; We construct a grid-like graph, where each pixel is the node of the graph, and each arc connects neighboring pixels. Two nodes are virtually added called Source(S) and Sink(T) and connected with infinite weight arcs to all pixels in the leftmost and right most columns in the image respectively.

  An S/T cut C is defined as a partitioning the nodes in the graph into disjoint subsets S and T. The cost of this cut is defined as the sum of the weights of the boundary arcs (p,q) where p $\in$ S and q $\in$ T.

Graph Cut for Seam Carving

Using the above defined formulation, we define an optimal seam as the cut with minimum cost among all valid cuts. A general cut is called valid, if it follows:

Monotonicity - The cut must include one and only pixel in each row

Connectivity - The pixels of the seams must be connected.

Having these conditions ensure that the retargeted image be free from imperfections.

- For Images: Every internal node($p_{i,j}$) is connected to its four neighbors(($p_{i,j+1}, p_{i,j-1}, p_{i-1,j-1}$, $p_{i+1,j-1}$). In horizontal forward (from S to T) direction, arcs weights are $E_1 = \mid \partial y(i,j) \mid + \mid \partial x(i,j) \mid = \mid I(i+1,j) - I(i,j) \mid + \mid I(i,j+1) - I(i,j) \mid$. For backward horizontal edges, the weights are ∞. This ensures the monotonicity of the obtained cut. To ensure connectivity, backward diagonal arcs are assigned ∞ weights. This is called backward energy function Figure 1.

- For Videos: The extension of the approach to the video For vertical seams, consider $X \times T$ planes and use same graph construction as in $X \times Y$, including backward diagonal ∞ arcs. Source and Sink are connected all leftmost and rightmost nodes of all frames respectively. The obtained partition will be monotonic in both time and space, due to horizontal constraints in each frame.

- Forward Energy Figure 3 - The backward energy and similar energy functions when used for video retargeting, causes artifacts in the frames. This is due to added energy when seam is removed. The energy is introduced due to new edges created by previously non adjacent pixels that become neighbors when the seam is removed. To correct for this, Rubinstein et. al.[5] proposed forward energy formulation of the graph cut, which finds seam that minimizes the inserted energy after its removal instead of backward at the image before removing the seam. These seams might not be minimal energy cuts but minimum energy is added, hence artifacts are very less in the retargeted image. To define the forward energy cost in graph cut, we need to create a graph whose arc weights will define the cost of the pixel removal according to the three possible seam directions. A new horizontal edge is added from I(i,j-1) $\rightarrow$ I(i,j+1) in all three cases as $p_{i,j}$ is removed. Hence, $+LR = \mid I(i,j+1) - I(i,j-1) \mid$ is assigned to arc from left to right neighbors. This accounts for energy added by removing horizontal edges. Similarly, we need to account for the cuts in vertical pixel edges. We assign $+LU = \mid I(i-1,j) - I(i,j-1) \mid$ i.e. Left-Up to the upward vertical arc between $p_{i,j}$ and $p_{i-1,j}$ and weight $-LU = \mid I(i+1,j) - I(i,j-1) \mid$ to the downward vertical arc between $p_{i,j}$ and $p_{i+1,j}$.

  For video, for vertical seams, intersection of every slice on the $(X \times T)$ dimension with the seam creates a 1-D seam on the plane. Hence, the same formulation on $X \times T$ as $X \times Y$ works. So, a cost is introduced by the pixel-temporal edge when a pixel is removed. Using the similar idea as X×Y, appropriate seams are found.

Results

I have computed the results using the above mentioned algorithm. The code for image retargeting takes around 20 seconds per seam, it overall took around 15-20 mins for each of these image. The implementation completely uses PyMaxflow[4].

The following are some observations based on the output received:

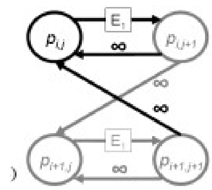


Figure 1: Backward Energy Cost Function for Graph cut Seam carving

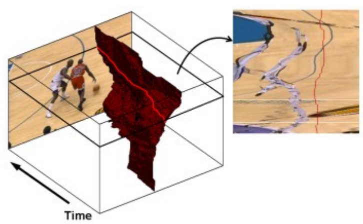


Figure 2: Graph Cut for Video

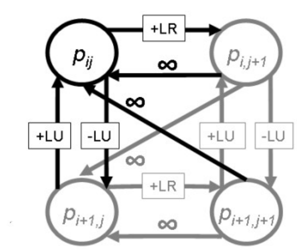


Figure 3: Forward Energy for Graph Cut Seam Carving

1. The results for Figure 1(a) are a bit squeezed up. This perfectly depicts one of the limitations of the forward energy mentioned in the paper. Forward energy tried to protect the structure of the image, but can resize important objects in a non pleasing way. The peacock in the image is kind of squeezed, the structure of the objects in the image are preserved but are reshaped terribly.

2. The results for Figure 2(a) are also disastrous. This is due to the net in the background(texture in background). Although it is not important but using forward energy, none of the seams aren't passing through that region. I tried to resolve this one by blurring the gray image which I am using to find the seam for the mask. The results for the same can be observed in Figure:5. Also, one can observe as we increase the blur kernel size, the background is much suppressed and seams find their way from the actual background instead of the man.

3. In the results of Figure 3(a), and Figure 4(a), we can clearly observe that the text and numbers are not much reshaped and are still intact in the retargeted image, this is due to structure preserving property of Forward Energy.

For the results of the video retargeting, due to lack of computational power, I had to resize each frame of the input video by a factor of 0.5 to be executed in possible time. The resized input took nearly 3-4 hours with the computational power of Google Colab. I list my observations/explanations about the output as follows:

- Using Image Retargeting method on each of the video frames and compiling each of them back into the video: The results in both video is very jittery, the frames get wavy, because the local minimum of the forward energy can be very different over time. This creates such a disturbance in the video.

- Using static seams: The static seams performs better in the rat video as compared to the golf video. In the golf video, the ball gets disappeared in the later part. This is due to the fact the seam isn't adapting with time and is static.

- Using video as 3D cube and find 2D manifold seams: This technique gives best results for the videos, the objects are resized very properly and all of them are present.

Further Extensions

For the videos extension part, for the "*vid_cam_motion.mp*4", we can split the video with the time segments where the scenary changes completely. The partition of video can be done using detection of SIFT features in each frame, time where most of the features changes abruptly should be considered a new scene. Further we can apply seam carving independently to each of these partitions to obtain a retargeted video.

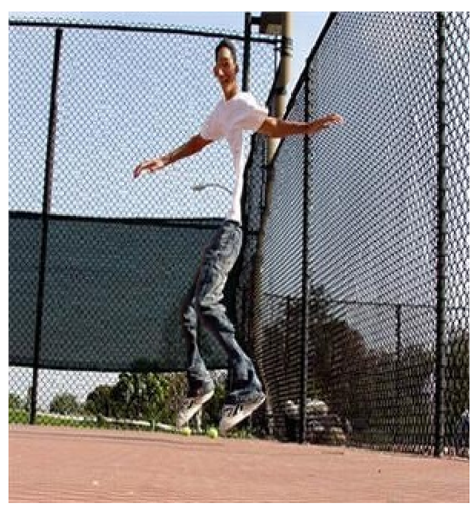
For the "*highway.mov*", we should detect objects which are entering and leaving the scene, segment them out, maintaining the background as it(we can interpolate from the neighboring frames). and retarget each of them independently. As the objects are moving in the video frames, they should not be cropped out, or basically not much of their seams should be removed. And while retargeting the segmented objects, we can record the seams and carve the corresponding seams out of the main video. This technique will be highly dependent on the background of the image.

1(a) Original Image

1(b) Retargeted Image

Figure 4: From 336x400 to 336x300

2(a) Original Image



2(b) Retargeted Image

Figure 5: From 320x400 to 320x300

3(a) Original Image

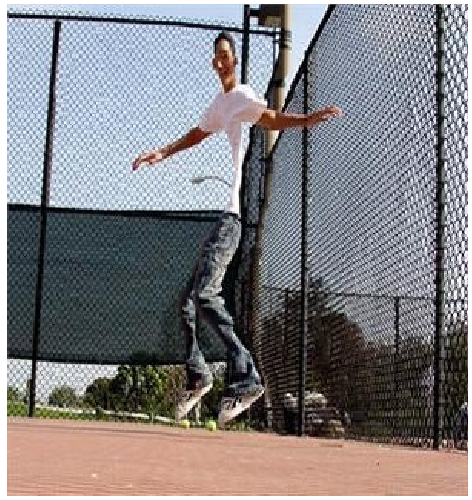
3(b) Retargeted Image

Figure 6: From 768x512 to 768x384

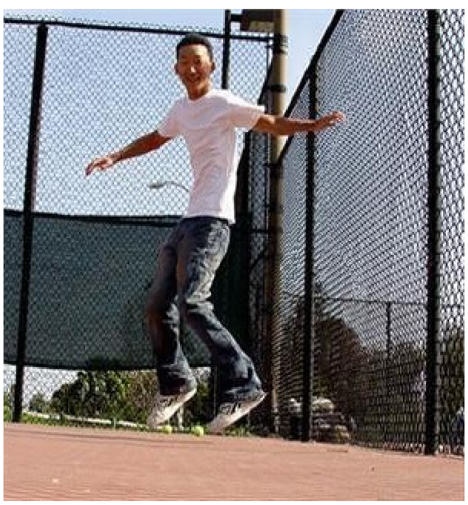
4(a) Original Image 4(b) Retargeted Image

Figure 7:

5(a) Original Image



5(b) Retargeted Image without blur



5(c) Retargeted Image with Gaussian Blur ksize 5

5(d) Retargeted Image with Gaussian Blur ksize 7

Figure 8:

References

[1] Shai Avidan and Ariel Shamir. Seam carving for content-aware image resizing. In *ACM SIGGRAPH 2007 Papers*, SIGGRAPH '07, page 10–es, New York, NY, USA, 2007. Association for Computing Machinery.

[2] Laurent Itti, Christof Koch, and Ernst Niebur. A model of saliency-based visual attention for rapid scene analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(11):1254–1259, 1998. Funding Information: This research was supported by the U.S. National Science Foundation, the Center for Neuromorphic Engineering at Caltech, and the U.S. Office of Naval Research.

[3] Feng Liu and Michael Gleicher. Automatic image retargeting with fisheye-view warping. In *Proceedings of the 18th Annual ACM Symposium on User Interface Software and Technology*, UIST '05, page 153–162, New York, NY, USA, 2005. Association for Computing Machinery.

[4] PNP productions. Seam carving tutorial, 2013.

[5] Michael Rubinstein, Ariel Shamir, and Shai Avidan. Improved seam carving for video retargeting. *ACM Transactions on Graphics (SIGGRAPH)*, 27(3):1–9, 2008.

[6] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001*, volume 1, pages I–I, 2001.