Improved Seam Carving for Video Retargeting - Sned Noodles

Date - 19th Nov, 2020

Contents

1.Team Members and Roll No.

Team Member Name	Team Member Roll No.
Aakash Dantre	2018101039
Jay Sharma	2018101033
Dhurv Arya	2018101003
Varun Chhangani	2019121011
Vishal Verma	2018101072

2.Overview

2.a. About seam carving

Seam: monotonic and connected path of pixels going from the top of the image to the bottom, or from left to right. Satisfying the following constraints:
➢ Monotonicity: the seam must include one and only one pixel in each row (or column for horizontal seams).
➢ Connectivity: the pixels of the seams must be connected

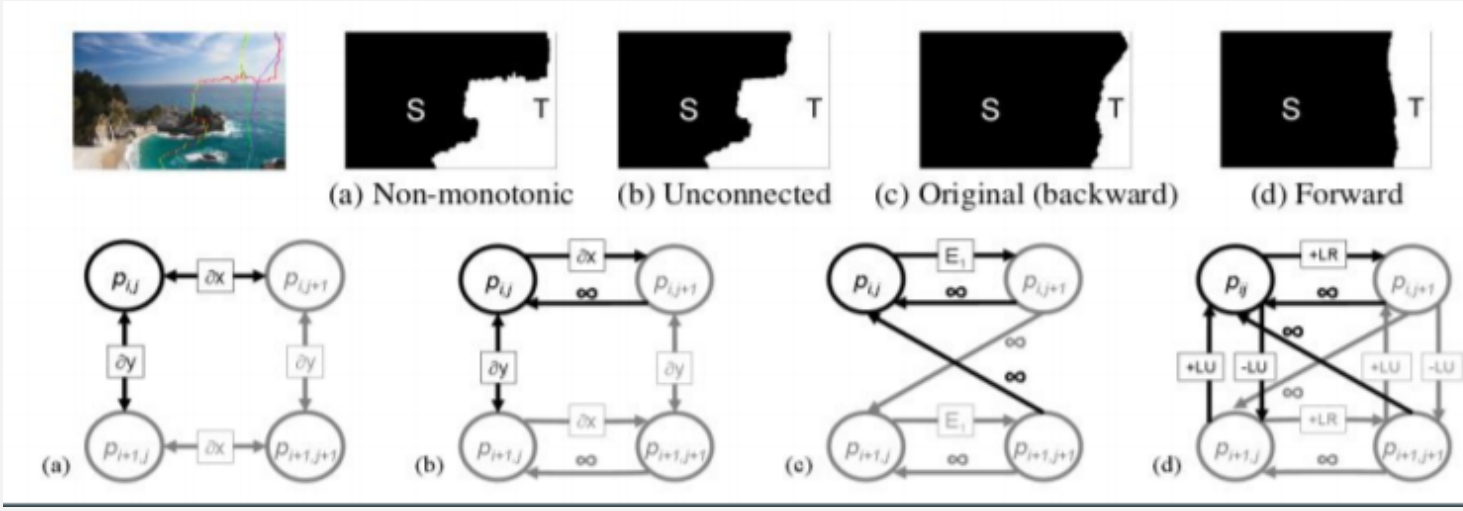
Seam carving is an effective technique for content aware image retargeting. Video, like images, should support content aware resizing. Instead of removing 1D seams from 2D images **we remove 2D seam manifolds from 3D space-time volumes**.

2.b. About Graph cut

To achieve this we use **graph cuts** that are suitable for 3D volumes.

We formulate the seam carving operator as a minimum cost graph cut problem on images and then extend this formulation to videos.

❖ Each pixel is considered a node and arc (edges in graph terminology) is drawn between neighbouring nodes.
❖ Virtual terminal nodes, S (source) and T (sink) are created and connected with infinite weight arcs to all pixels of the leftmost and rightmost columns of the image respectively.
❖ The optimal seam is defined by the minimum cut which is the cut that has the minimum cost among all valid cuts.



2.c. Graph cut for videos

➔ Consider the X × T planes in the video cube and use the same graph construction as in X × Y including backward diagonal infinity arcs for connectivity.
➔ A partitioning of the 3D video volume to source and sink using graph cut will define a manifold inside the 3D domain
➔ The graph cut algorithm runs in polynomial time, but in practice was observed to have linear running time on average [Boykov and Kolmogorov 2004].
➔ The graph cut approach to seam carving allows us to extend the benefits of content-aware resizing to video.

3. Design Constructs

3.1 Narrative

Here, we examine the classes and files used.

Narrative

Remove_seam.py is first run, which opens the video file and array it such that it comes in 3d volume. It further store that in array.txt file and runs a bat file(flow.bat) or bash file(bash flow.sh) on the basis of your OS. These files drive the rest of code. Preproc.cpp is executed which calculates graph edge based energy functions and stores it. The output of it is used by graphcut.exe which is compiled using main.cpp, graph.h, block.h, graph.cpp, maxflow.cpp. All the major seam calculations happens in here, as it gives seamout.txt as the final output. Get_seamtry.py uses seamout.txt and remove all the seams from our 3d volume of video and stores it back in array.txt. Visual_compare.py coverts array.txt to video format.

3.Responsibility(ies) of each major file

Responsibility(ies) of each major file

Classes	Description
remove_seam.py	- Driver, first file to be run - Coverts video, resize it according to **config.py** - Calls **flow.bat or bash_flow.sh** - Remove **NUM_SEAMS** number of seams from video.
flow.bat or bash flow.sh	- Drives the rest of code - Removes one seam from video - Compiles preproc.cpp to create **preproc.exe** - Compilesmain.cpp, graph.h, block.h, graph.cpp, maxflow.cpp and create executable **graphcut.exe**.
preproc.cpp	- Uses **energy function** - Creates edge based graph - Output of this file is passed on to graphcut.exe
block.h	- Template classes Block and DBlock - Implement adding and deleting items of the same type in blocks. - If there there are many items then using Block or DBlock is more efficient than using 'new' and 'delete' both in terms of memory and

	time
graph.h	- Helper function for graph.cpp to compute graph cuts
graph.cpp	- it is mentioned DP isn't enough and graph cuts are needed for video, - Even after that, the tradition graph cut algorithms take O(E*V^3) or O(E*V^2) time depending on use of adjacency list or matrix. - The implemented graph cut algorithm runs in o(V*V) time complexity
main.cpp	
maxflow.cpp	- Functions for processing active list. - i->next points to the next node in the list - There are two queues. Active nodes are added to the end of the second queue and read fromthe front of the first queue. If the first queueis empty, it is replaced by the second queue (and the second queue becomes empty)
get_seamtry.py	- seam removed array is created using **seamout.txt** and stored in array.txt
visualcompare.py	- Converts **array.txt** into a comparison video of before and after videos
Config.py	- Have constant values such as resolutions, path of videos, **NUM_SEAMS**
exportToVideo.py	- export the comparison video as mp4 video file
rgbVidMaker.py	- export the final processed RGB video as mp4 file

4. Final Output

4.1 input

4.2 output

5. Installation

5.1 prerequisites

- g++ for c++ compilation
- python3
- numpy
- opencv-python

5.2 running

Install code from github

- git clone https://github.com/Digital-Image-Processing-IIITH/project-sned-noodles.git

- cd src
- bash compile.sh (or compile.bat for windows users) (necessary for setup)
- vim config.py (edit the configuration as per demand)

- python remove_seams.py (process video and remove seams)
- python visualcompare.py (view side by side comparison of processed and original video)
- python exportToVideo.py (export the comparison video as mp4 video file)
- python rgbVidMaker.py (export the final processed RGB video as mp4 file)