Advanced Computer Graphics Final Project

Joaquín Ruales (jar2262)

December 17, 2014

Progress since my final presentation

After my final presentation on Monday, I have extended my algorithm so that it also works with video textures, not only image textures. Below I describe my implementation and results. Additionally, I have made it possible to place patches anywhere now, even if the patch overflows past any edge, and I crop the patch so that it fits into the output image before actually placing it. My most impressive results were the metallic nuts image, and with the flame video.

I have not managed to implement the "Entire patch matching" patch placement (which uses FFT and integral images for speed), but this would be the logical next part to implement, and it would provide an even nicer result, since it makes sure that we place patches in places with a good match, not just anywhere.

One thing I would like to work on outside from the class is to try connecting pixels in the graphcut to diagonal neighbors in addition to horizontal and vertical neighbors. I believe that this would bring better results (albeit with some extra computational cost), since it would avoid having a bias in favor of cutting long axis aligned lines.

Implementation

For this project, I have implemented parts of the paper "Graphcut Textures: Image and Video Synthesis Using Graph Cuts."

In my paper presentation for the class (Oct 1), I presented an overview of the aforementioned paper. This presentation is in the file "paper_presentation.ppsx". Furthermore, I presented about my implementation of the paper in my final presentation (Dec 15). This presentation is in the file "final_presentation.ppsx".

The paper covers the following topics:

- Image texture synthesis
  - Patch placement (Figuring out where to put the overlaid patch, i.e. finding the offset)
    - Random placement

    - Entire patch matching
    - Sub-patch matching
  - Seam finding (Figuring out where to cut the overlaid patch so that it blends as well as possible with the rest of the synthesized image)
  - Optimizations for patch placement (FFT and integral images)
  - Improvements for seam-finding
  - Video texture synthesis
    - temporal
    - spatio-temporal
    - seamless, infinite looping videos
- Interactive Image Synthesis (Image photo-montages)

Out of these, I have implemented texture synthesis for both images and videos, but using user input for the patch placement instead of the algorithms proposed in the paper.

For image texture synthesis (GraphCutTextures.m), the first texture patch is placed at the top-left corner. After that, at each iteration, the user program asks the user to click on the currently generated image in order to select the position to put the next patch (top-left corner), and places it for the user to see. Each patch should be placed so that it touches both the black region and the previous image (although we saw during my presentation that it also works when patches are places entirely in the black background).

For video texture synthesis (VideoGraphCutTextures.m), the first texture parch is placed at the top-left corner starting at timeframe zero, and the second patch is placed at a hard coded 3d position that can be changed directly in the code. In addition to extending the length of the Flame video (see Results section), I also made a seamless, infinitely looping video of the flame, and I saved it as a GIF in "video_results/flame_result_repeated.gif".

Code

I wrote my program in MATLAB. The program is self-contained (assuming you have the Image Processing toolboxes installed in your MATLAB). The only external library that it uses (A library that computes min cuts) is included, and I have acknowledged its authors in the Acknowledgements section of this document.

My main code is in:

- Bk_matlabGraphCutTextures.m
- Bk_matlabVideoGraphCutTextures.m

Where GraphCutTextures.m works with images and VideoGraphCutTextures.m works with videos. I also have some experimental code (not worth looking at) in:

- Bk_matlabVideoGraphCutTexturesFountain.m
- Bk_matlabVideo.m

Results

My image synthesis results are available under the "image_texture_results" folder next to this document.

My video synthesis results are available as a playlist on YouTube: `https://www.youtube.com/playlist?list=PL0FOk__4FY_NASTu5TuKxyF_LGVOfMa9X`

Acknowledgements

For this project, I have used an external graph cut C library (BK max flow algorithm) implemented by Vladimir Kolmogorov, with MATLAB bindings by Andrew Delong. I found this library (and MATLAB bindings) on the following website: `http://vision.csd.uwo.ca/code/`

This library is based on the paper:

"An Experimental Comparison of Min-Cut/Max-Flow Algorithms for Energy Minimization in Vision." Yuri Boykov and Vladimir Kolmogorov. In IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI), September 2004