SEAMLESSLY CREATING GIFS FROM A VIDEO CLIP WITH TEXTURE VARIETY PRESERVATION

KUNJAL PANCHAL [32126469]
CMP SCI 670 COMPUTER VISION BY SHUBHRANSU MAJI
COLLEGE OF INFORMATION & COMPUTER SCIENCES,
UMASS AMHERST

ABSTRACT

In current age of technology, GIFs[Graphics Interchange Format] are popular because of their relatively smaller size compared to other formats. Moreover, loading images online is quicker without losing its quality. And they simply convey the message better than a static image. In the paper "Video Textures" by Schodl, Szeliski, and Essa; they created a "new type of medium" called a video texture, which is a continuous infinitely varying stream of images, same concept as a GIF.

They presented techniques for analyzing a video clip to extract its structure, and for synthesizing a new, similar looking video of arbitrary length, which is seamless. The goal is that the method needs only few frames out of the whole clip, it works infinitely continuously with varying patterns to give an illusion of arbitrarily long video clip. We implement this method from scratch.

But some clips like water, fire, grass in wind, candle flames, flags face some jitter (conspicuous jumps between frames)paper mentions this can solved by cross-fading, but they have not implemented it, we will, in this project, try to remove all the jitters; this is easily fixed by modifying Distance Matrix D so that the pairwise distance between frames also considers a few frames around it. This can be achieved by filtering D using a length 2 or 4 filter with weights set to the binomial coefficients (to approximate a Gaussian distribution, with the correct width). Next, in video clips featuring time lapses, the lighting changes too drastically when we choose only few frames from many. This can be solved by some pre-processing – normalizing intensities.

APPRAOCH AND ALGORITHM ANALYSIS

There are two main parts to the algorithm. First, the video texture must be extracted from the input video. Second, a new video that synthesizes the video texture must be created. But, the key idea behind the algorithm is this: given a frame of a video, we can select a plausible next frame by randomly picking a similar frame to the one that would have been played in the original video. This next frame may not be the actual next frame in the input, but it may be. In this way, we can innitely and smoothly extend a video.

EXTRACTING THE VIDEO TEXTURE

To extract video textures, we need to compute how similar pairs of frames are to each other. This can be done by calculating the SSD between each pair of frames, and storing the results in a distance matrix, D. From there, we can compute a probability matrix, P, which assigns probabilities between pairs of frames. P can then be used to calculate the next frame in the output video given a current frame: since a frame is a row, we discretely sample the next frame using the probability distribution across a row of P. P is created using an exponential function and dividing by a constant, sigma. The paper notes that sigma is (often, but in our case always) set to a small multiple of the average non-zero D

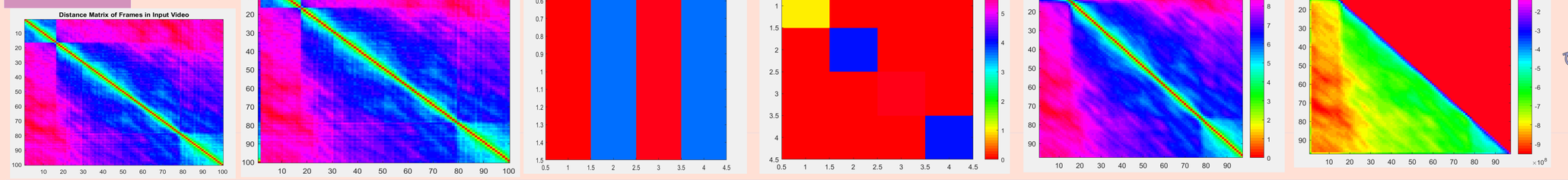
PRESERVING DYNAMICS

In some cases, the input video has a fluid motion that the video texture should preserve. The paper gives the example of a pendulum swinging: the algorithm described above doesn't account for the fact that original video has a side-to-side motion. So, the resulting texture may jitter back and forth since there's no distinction that the next frame may have come from the left side or the right side in the original video. This is easily fixed by modifying D so that the pairwise distance between frames also considers a few frames around it. This can be achieved by ltering D using a length 2 or 4 lter with weights set to the binomial coecients (to approximate a Gaussian distribution, with the correct width).

WRITING VIDEO LOOPS

The motivation behind video loops is to preselect a sequence of loops that can be repeated, so that the resulting texture smoothly repeats when played on a conventional video player's repeat setting. We attempted to implement video loops, but were unable to produce any compelling results with the algorithm. The paper gives an example to describe how the video loops dynamic programming algorithm and scheduling algorithm works. Our implementation was successful on this example, but when we tested on real videos there were very obvious skips (see failure cases below) that shouldn't occur. We feel that the most likely cause of the problem was our selection of primitive loops.

RESULTS



REFERENCES Schödl, Arno & Szeliski, Richard & Salesin, David & Essa, Irfan. (2000). Video Textures. Proc. of ACM SIGGRAPH. 2000. 10.1145/344779.345012.