UNIVERSIDADE DE SÃO PAULO
INSTITUTO DE CIÊNCIAS MATEMÁTICAS E DE COMPUTAÇÃO
Departamento de Ciências de Computação

Digital Image Processing

Eulerian Video Magnification - Final Report

Gabriel Scalet Bicalho, 8937204

Gustavo Ferreira Ceccon, 8936822

Victor Moneratto Saia, 8937009

July 30th, 2017

Objective

We aim to reveal temporal variations in videos that are difficult to see with the naked eye and display them in an indicative manner. The method we chose is called Eulerian Video Magnification, it takes a video sequence as input and applies spatial decomposition followed by temporal filtering to the frames. The resulting signal is amplified to reveal hidden information. Using this method, we are able to amplify and reveal small motions.

Input

Essentially any video can be explored by this method, we can easily capture from a webcam, a cellphone camera or download royalty-free videos from the internet. For demonstration purposes, selected a few videos which can be found here.

Algorithm

Capture a Video

The video's size doesn't need to be power of 2, but it is recommended to make sure the image can be resizable **n** times by the pyramid method.

Apply Blur or Edge Finding Filtering (Optional)

Apply gaussian blur or some edge finding filtering, like sobel filter method etc.

Downsample Image Using Image Pyramids

Downsample the image **n** times using either a Gaussian or Laplacian filter, using *pyrDown* method. The image is blurred when downsampled.

Apply a Frequency Filtering Method

A Fourier Transform analysis and clamping can be used to filter the frequencies or a Low Pass Filter (LPF), using Infinite Impulse Response (IIR). The difference is subtle, even in performance, since the FFT can be parallelized using threads. The IIR needs only the previous frame and the current one, but needs to be done sequentially.

1. FFT

Using scipy’s FFT, the algorithm can be parallelizable and cacheable. The history buffer needs to be maintained and updated, increasing memory usage. Then we just cut all the frequencies below and higher than the frequencies specified by the parameters. The amplification is applied in the lower pyramid level.

1.1 Append Image to a Circular History Buffer

1.2 Apply FFT to the History Buffer

1.3 Apply Frequency Filtering to Extract a Range of Interest

2. Infinite Impulse Response (IIR)

Two copies of the matrices needs to be made, which holds the upper bound frequencies and the lower bound frequencies. The parameters r1 and r2 are calculated using $\alpha = dt / (RC + dt)$, where $RC = 1 / (2\pi fc)$ and fc is the frequency cutoff (upper and lower). Then we apply a linear interpolation between the previous frame (filtered) and the current frame (not filtered). The amplification is made in some pyramids levels, depending the *lambda* and *alpha* factor and the algorithm specified by the paper.

2.1 Calculate the Constants

2.2 Interpolate Previous Filtered Frame and the Current Frame

2.3 Subtract the Upper LPF by the Lower LPF

2.4 Additional Filtering by Applying Other Multiplications

Create *Motion Maps* with the Filtering Method

The *motion map* is returned by the Fourier Transform method using the inverse function IFFT. The IIR method needs to subtract the upper frequencies by the lower frequencies.

Upsample and Intensify *Motion Maps*

The multiplication of the exaggeration (amplification) factor depends on the filtering method. The upsample is made using pyrUp method.

Compose and Display the Resulting Image

Write the frames to the output video.

Running the Code (Demo)

The algorithm is in `python 3.6`, and requires the libraries `pyfftw` and `scipy`, which can be installed using `pip` or found at http://www.lfd.uci.edu/~gohlke/pythonlibs/.

After the libraries and its dependencies are installed, the program can be run as follows:

```
python3 Eulerian.py input_file output_file amplification min_frequency max_frequency number_of_pyramids
```

Where

- `input_file` is the file to be processed.
- `output_file` is the output file destination. It *must* be an `.avi`. The target directory must exist for the file to be successfully created.
- `amplification` is the rate at which the detected motions will be amplified.
- `min_frequency` is the lowest frequency of movement the algorithm will apply amplification to.
- `max_frequency` is the highest frequency of movement the algorithm will apply amplification to.
- `number_of_pyramids`is the number of times the image will be halved in size before processing. A higher number increases the speed of processing and reduces noise, but lowers the sensitivity to movement.

For the inputs used as example, the following commands were used:

```
python Eulerian.py in\face.mp4 out\face_4.avi 120 0.8 1.1 4
python Eulerian.py in\face.mp4 out\face_2.avi 120 0.8 1.1 2
python Eulerian.py in\baby.mp4 out\baby.avi 80 0.4 1.4 2
python Eulerian.py in\baby2.mp4 out\baby2.avi 200 2.3 2.7 2
python Eulerian.py in\wrist.mp4 out\wrist.avi 60 0.4 1.2 2
python Eulerian.py in\input_05.mp4 out\input_05.avi 32 0.6 1.0 2
```

Results

As it can be seen in the results, the results proved themselves to be satisfactory, increasing the change in both color and movement. One unfortunate side-effect from the algorithm is that appropriate parameters can be difficult to find and reaching an acceptable result requires testing on a case-by-case basis.

We should also note that a similar result could be achieved with a IIR mentioned above instead of a Fourier Transform. However, it is harder to implement, so for this project we chose to stay with the Fourier Transform to allow a faster development of the Demo program.

References

- Video Magnification - MIT
- Eulerian Video Magnification (source and examples) - MIT
- Wu, Hao-Yu. Rubinstein, M. Shih, E., Guttag, J., Durand, F. and Freeman, W. Eulerian Video Magnification for Revealing Subtle Changes in the World