Malfunctioning engine clamping components detection using Phase-based video magnification

Anuar Maratkhan[1], Kudaibergen Urinbayev[2], and Ibrakhim Ilyassov[1]

[1]Department of Computer Science
[2]Department of Robotics and Mechatronics
[1,2]Nazarbayev University
53 Qabanbay Batyr Ave., 010000
Astana, Kazakhstan
Email: {anuar.maratkhan, kudaibergen.urinbayev, ibrakhim.ilyassov}@nu.edu.kz

***Abstract*—Human visual system has some limitations. While human can recognize some vivid oscillations such as leave swings on the wind, some of subtle fluctuations like pulse or sway of bridge left unperceivable for the naked eye. However, small changes in motion could contain essential information of the system and could be quantitively and qualitatively analyzed. These analysis are done through a video magnification that we are applying for engine mount inspection in our work.**

I. Introduction

Human visual system has some limitations. While human can recognize some vivid oscillations such as leave swings on the wind, some of subtle fluctuations like pulse or sway of bridge left unperceivable for the naked eye. However, small changes in motion could contain essential information of the system and could be quantitively and qualitatively analyzed. These analysis are done through a video magnification that we are applying for engine mount inspection in our work.

Several video magnification techniques were proposed. In 2005, Lie et al suggested Lagrangian approach to amplify motions where pixels are tracked and motion vectors are amplified directly to synthesize videos with larger motions. This method required motion segmentation and manual correction of filling the holes after video processing [1].

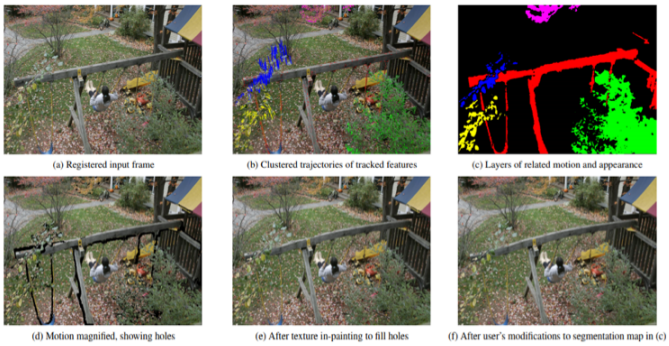


Fig. 1. Summary of motion magnification processing steps [1]

Further, in 2012 a new magnification method was proposed that used Eulerian approach. Eulerian Video Magnification (EVM) computational method of a video amplification of subtle motions and color changes over time [2]. The main feature of this method is that instead of tracking and shifting each pixel explicitly as in Lagrangian approach Eulearian method operates with the color change of the pixel in the temporal domain. Therefore, due to EVM subtle color changes as well as the slight motions could be magnified.

The figure below describes the working principle of the EVM algorithm. Firstly, the video goes through the spatial decomposition to obtain spatial frequency bandwidth using Laplacian pyramids to decrease noise and allow to operate with small structures. The band-pass filter then applied to remove noise and inappropriate artefacts. Each pixel goes through temporal processing where the intensity is magnified by some factor. Afterwards video goes through reconstruction.

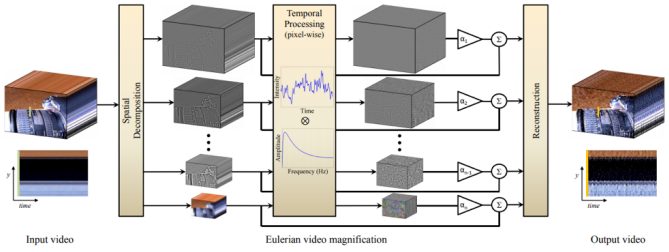


Fig. 2. Overview of the Eulerian video magnification framework. The system first decomposes the input video sequence into different spatial frequency bands, and applies the same temporal filter to all bands. The filtered spatial bands are then amplified by a given factor , added back to the original signal, and collapsed to generate the output video. The choice of temporal filter and amplification factors can be tuned to support different applications. For example, we use the system to reveal unseen motions of a Digital SLR camera, caused by the flipping mirror during a photo burst (camera; full sequences are available in the supplemental video). Referenced from [2]

Thus, the change in the pixel intensity allows to amplify motion in spatial domain.

However, EVM method has its own drawback due to the

linearity: noises are amplified gradually with motion magnification as well as some visual artefacts become visible in a fields of high frequency motions.

II. Phase-based Video Magnification

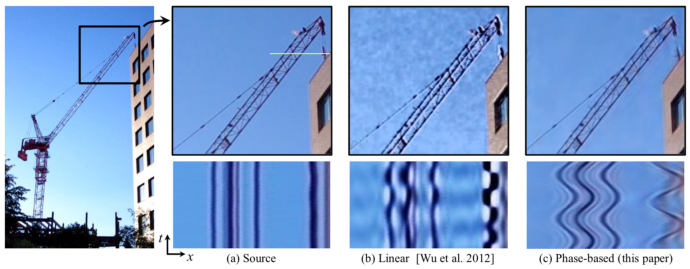


Fig. 3. Motion magnification of a crane imperceptibly swaying in the wind. (a) Top: a zoom-in onto a patch in the original sequence (crane) shown on the left. Bottom: a spatiotemporal XT slice of the video along the profile marked on the zoomed-in patch. (b-c) Linear [Wu et al. 2012] and phase-based motion magnification results, respectively, shown for the corresponding patch and spatiotemporal slice as in (a). The previous, linear method visualizes the cranes motion, but amplifies both signal and noise and introduces artifacts for higher spatial frequencies and larger motions, shown by the clipped intensities (bright pixels) in (b). In comparison, our new phase-based method supports larger magnification factors with significantly fewer artifacts and less noise (c). The full sequences are available in the supplemental video. Referenced from [3]

In the Fourier transform domain the amplitude of the sinusoids depends on amplitude of the basis function, while the change in phase corresponds to translation of the sinusoid. However, Fourier shift theorem works on the global motion, whereas in most cases only certain motions need to be magnified. Therefore, concept of localized Fourier series is introduced and it could be explained by Complex Steerable Pyramids.

A. Complex Steerable Pyramids

The steerable pyramid [5] is an overcomplete transform that decomposes an image according to spatial scale, orientation, and position. We understand this work as a way to do local Fourier transform for certain bandpassed frequencies.

The steerable pyramid [5] is an overcomplete transform that decomposes an image according to spatial scale, orientation, and position. We understand this work as a way to do local Fourier transform for certain bandpassed frequencies. The steerable pyramid method mentioned above is used to measure local amplitude and phase, which are exploit in [3] to process motion. Those phases are then magnified by . These works are replicated in our work for further use in engine mount test through video magnification that is unseen to the naked eye.

Figure above demonstrates the working principle of the Phase Based video magnification method. The main difference between just mentioned method and EVM is that the in video processing.

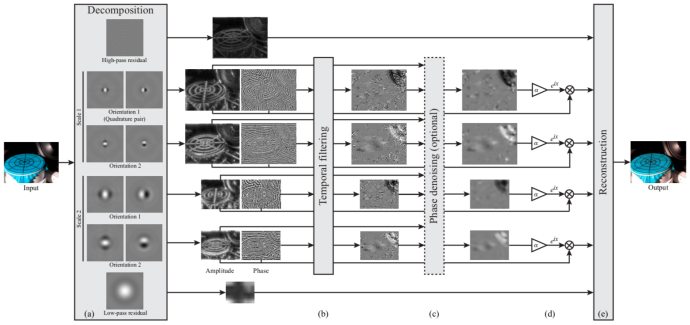


Fig. 4. Our phase-based approach manipulates motion in videos by analyzing the signals of local phase over time in different spatial scales and orientations. We use complex steerable pyramids to decompose the video and separate the amplitude of the local wavelets from their phase (a). We then temporally filter the phases independently at at each location, orientation and scale (b). Optionally, we apply amplitude-weighted spatial smoothing (c, Sect. 3.4) to increase the phase SNR, which we empirically found to improve the results. We then amplify or attenuate the temporally-bandpassed phases (d), and reconstruct the video (e). This example shows the processing pipeline for the membrane sequence (Sect. 4), using a pyramid of two scales and two orientations (the relative difference in size between the pyramid levels is smaller in this figure for clarity of the visualization). Referenced from [3]

III. Experimental setup

To test given method the video of car engine was filmed. The video was shoot with the 25 fps Canon 550D. Due to the limited resources and time, video of car engine of Toyota Yaris (manufactured in 2011) was taken outdoor in cloudy/windy weather condition.

Since car mount deinstallation and reinstallation required special tools and some mechanical engineering skills it was decided to remove one of the spark plugs because according to https://www.carsdirect.com/car-repair/5-causes-of-engine-vibration worn out spark plugs is a common reason of the motor vibrations.

IV. Usage

We divide the work done into two parts (MATLAB and Python) because codes are separately written in MATLAB and Python programming languages. The MATLAB part is taken from original work of [3], and is used to magnify the video. On the other hand, the second part uses the output video from the first part and compares the video of properly working engine with the video of engine with defects in engine's mounts. The both parts are described in detail below.

A. MATLAB code

The original MATLAB code from [3] has been used to magnify the video of working car engine. The proper usage of the part is described below.

1) Input videos: To add videos that are going to be magnified, you need to add video files inside `input_files` directory inside main `MATLAB` directory. Further, the videos will be taken from this directory.

2) Processing the video: In order to magnify the chosen video, you need to run MATLAB function in the main MATLAB directory with the name `robt310_final_project`. The function accepts one argument - the name of the file (as a string). So, to run the video magnification method, call a function from terminal like this:

```
robt310_final_project('video.mp4');
```

In case you want to customize the function itself, change the function paramaters inside it. The parameters are set to process car engine, as in the example code of [3] work. Default setting that can be customized are:

- `samplingRate` - the rate the video frames will be sampled. Higher the rate, slower the video play will be. We have decreased the original 400 down to 120 sampling rate because the original work has been processing the video with 400 fps compared to our 25 fps.
- `loCutoff` and `hiCutoff` - the range of low and high frequencies (Hz) to be bandpassed for magnification. The default parameters `loCutoff` = 15, and `hiCutoff` = 25 has not been changed from the original code.
- `alpha` - α rate according to which the video will be magnified. Higher the alpha, more the video will be magnified. The original work magnified the video with α = 15. We are magnifying it with α = 50 in order to see the magnification more clearly.
- `sigma` - parameter that is used in the magnification. Changing the alpha will result in the magnification style. We set it to default 3 as in the original code.
- `pyrType` - default set is 'octave'. Another option to use is 'halfOctave' which is said to produce different result and uses more computational resources. The parameter comes from complex steerable pyramids and is out of scope of our work. Thus, we have used the default value 'octave'.

The only recommended change is the `samplingRate` parameter that changes the video look in time according to number of frames per second (fps) in the video to be processed.

3) Output videos: The output files will be saved in `output_files` directory. The names of the files are set in code by using the original input filename, and the parameters used to process the video.

B. Python code

1) Matching frames of videos: There exist a problem that two videos, original and defected one, can move with shifted frames. For example, first videos engine can go up and second videos engine can go down, so they are moving in different phases. Therefore, it will be wrong without initial matching of videos to calculate RMSE.

First of all, we cut a rectangular region with engine from video using special GUI. Cropped image is used as template in order to find the region with this template in both videos. Thus, we found the maximum upper deviation in both videos by using locations of detected regions based on provided template. The first frame, where is maximum upper deviation occurs, will be used in order to match with the first frame of the second video, which has maximum upper shift of engine. We have written a function `getFirst()` in `match.py` module that returns pair value of first frame and upper boundary of engine shift.

2) Calculating RMSE: After matching of two videos by frames, we calculated RMSE for every frame and get the mean of all values for every frame. Thus, the resulting RMSE is **7.49**.

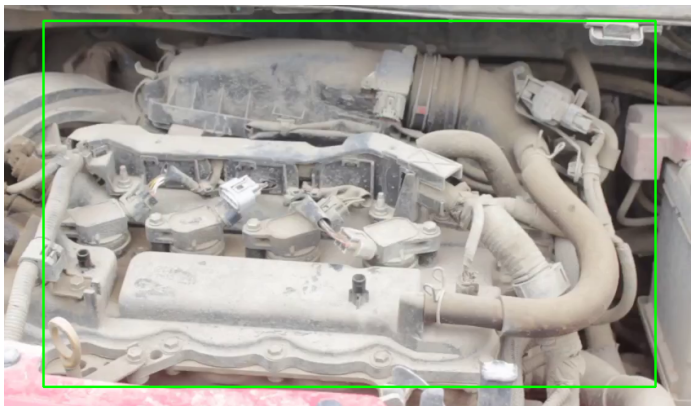
3) Usage for calculating RMSE:

a) Cropping a template image: Firstly, we need to crop a region with engine in order to match videos by frames. You need to call a python script and pass videos name as argument:

Example:

```
$ python crop.py engine.mp4
```

Secondly, select region by clicking and cropping using GUI:



Then, press c key in order to cut and save as template in roi.jpg file.

b) Calculating RMSE: Run python script and pass two videos filenames as arguments in order to compare them. Example:

```
$ python calculate.py 1.avi 2.avi
```

V. Results

We used phase based motion magnification and calculated RMSE for two videos: original and defected. Initially, RMSE of two videos was 7.62 without frame matching. The result with frame matching is 7.49. We could not identify the threshold in order to classify defected or not defected engines, because we have only one observation

VI. Conclusion

In conclusion, even though the video was taken with 25 fps the resulting processed video sequence shown considerable difference between two videos: with and without one spark plug. The output RMSE shown 7.49. In order to achieve more accurate results in video magnification it is highly suggested to use high FPS video camera because the video amplifies more steadily. In addition, to achieve less errors in video comparing indoor shooting is highly advised because even subtle camera shaking highly reflected in the processed file.

References

[1] C. Liu, A. Torralba, W. T. Freeman, F. Durand, and E. H. Adelson, "Motion magnification," *ACM Trans. Graph.*, vol. 24, no. 3, pp. 519–526, Jul. 2005. [Online]. Available: http://doi.acm.org/10.1145/1073204.1073223

[2] H.-Y. Wu, M. Rubinstein, E. Shih, J. Guttag, F. Durand, and W. Freeman, "Eulerian video magnification for revealing subtle changes in the world," *ACM Trans. Graph.*, vol. 31, no. 4, pp. 65:1–65:8, Jul. 2012. [Online]. Available: http://doi.acm.org/10.1145/2185520.2185561

[3] N. Wadhwa, M. Rubinstein, F. Durand, and W. T. Freeman, "Phase-based video motion processing," *ACM Trans. Graph.*, vol. 32, no. 4, pp. 80:1–80:10, Jul. 2013. [Online]. Available: http://doi.acm.org/10.1145/2461912.2461966

[4] B. Aubakir, B. Nurimbetov, I. Tursynbek, and H. A. Varol, "Vital sign monitoring utilizing eulerian video magnification and thermography," *2016 38th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, pp. 3527–3530, 2016.

[5] E. P. Simoncelli, W. T. Freeman, E. H. Adelson, and D. J. Heeger, "Shiftable multiscale transforms," *IEEE Transactions on Information Theory*, vol. 38, no. 2, pp. 587–607, March 1992.