

Accelerating Eulerian Video Magnification Using FPGA

Kening Zhang*, Xi Jin*, An Wu*

* Department of Physics, University of Science and Technology of China, Hefei, China

kning04@mail.ustc.edu.cn, jinxi@ustc.edu.cn, wuan@mail.ustc.edu.cn

Abstract—Video magnification can amplify and display the subtle motions that are impossible to see with our naked eyes. However, it's time consuming for CPU processing video magnification with high resolution. We propose a FPGA-based solution to Laplacian pyramid, temporal filter and pyramid reconstruction algorithm of the video magnification. The paper processes all pyramid levels and temporal filter with parallel and pipeline architecture. With the reconfigurable feature of FPGA, the temporal filter can be reconfigured which are all customizable by the user. We implement the whole video magnification system on Xilinx Kintex-7 FPGA board. When we process 1920×1080 video, the result shows that the hardware system is obviously speedup versus Eulerian Video Magnification on Intel i5 core.

Keywords—Eulerian Video Magnification; FPGA; Laplacian pyramid; subtle motions; motion magnification

I. INTRODUCTION

Because our visual system has limited spatiotemporal sensitivity, some small variations that have useful information are difficult to see with our naked eyes. In the real work, the small motions could contain important information and some will be useful in medical diagnosis. For example, while the blood is circulating in body, the variations in skin color that cannot be see with our naked eyes are changing [1] [2]. This small variation can help doctors diagnose heart function.

In order to reveal the important information hided in small variations, we have to magnify the small variations to be visible with our naked eyes. There are two main approaches: Lagrangian perspective and Eulerian perspective. Liu et al. have developed a technique, called Motion Magnification, which acts like a microscope for motion in video sequences [3]. It depends on Lagrangian method that relies on accurate motion estimation, which is computationally expensive and difficult to make artifact-free, especially at region of occlusion boundaries and complicated motions. Hao-Yu Wu et al. combine spatial and temporal processing of video to amplify subtle variations that reveal important aspects of the world around us [4]. The method is called Eulerian Video Magnification.

Eulerian Video Magnification method not only magnifies color variation, but also can amplify slight motion. Eulerian Video Magnification method computes the change of the color value in the time serial and amplifies it in a given frequency band of interest. This leads to a multiscale approach to

magnify motion without tracking or motion estimation. However, the program in CPU is executed sequentially, so CPU is slow to process high resolution video and it is weak to handle the algorithm of high complexity such as Laplacian pyramid, and those make CPU often difficult to meet the needs of embedded system.

In the paper, we propose a design that accelerates the Eulerian Video Magnification method with hardware. The hardware resources mainly include Field Programmable Gate Array (FPGA), Application Specific Integrated Circuit (ASIC), Graphics Processing Unit (GPU), and Digital Signal Processor (DSP). ASIC is designed based on customer needs, to ensure that customers' requirements on performance, chip area, power consumption and confidentiality can be met as far as possible, but it takes a lot of time and the consequent high costs [5]. GPU performs much better in parallel computing, but its power consumption is relatively high [6]. FPGA allows users to implement their designs by configuring the corresponding logic functions which can be used as not only a logic chip design functional verification platform to shorten the design cycle and reduce risks from changing needs; but also a coprocessor or a dedicated processor with high performance [5] [7]. And what makes FPGA even better is its low power consumption and cost.

The paper is organized as follows. This section introduces the importance and perspective of motion magnification and the methods of hardware accelerating. In the next Section II, the paper briefly introduces the Eulerian Video Magnification algorithm, and discusses the bottlenecks of the Eulerian Video Magnification. Then the details of FPGA implementations for Eulerian Video Magnification will be demonstrated in Section III. To verify the FPGA design, we test our Eulerian Video Magnification system and compare with the CPU using the same database in Section IV. Section V concludes the paper.

II. EULERIAN VIDEO MAGNIFICATION ALGORITHM

The paper studies the pixel value variation over time using Eulerian Video Magnification method and amplify the subtle signal to be visible which is inspired by Eulerian perspective.

The Eulerian Video Magnification algorithm combines spatial and temporal processing to reveal subtle variation in a video. We amplify the subtle variations over time in a multi-scale space. Eulerian Video Magnification that reveals the

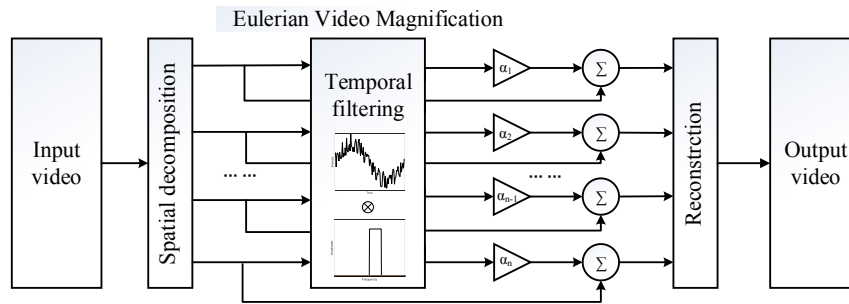


Figure 1. Overview of the Eulerian Video Magnification framework

subtle motions doesn't track or estimate motion like Lagrangian methods [2], but rather amplifies the temporal changes for the whole pixels combining spatial and temporal processing. This method relies on the same differential approximations that form the basis of optical flow algorithm [8] [9]. It is illustrated in Figure 1. Spatial decomposition decomposes every video frame into different frequency bands. Image noise will be produced by the sensor and circuitry of a scanner or digital camera. When we only amplify the video to reveal the information of interest, we have to suppress the noise to increase the Signal-to-Noise Ratio (SNR). So we take spatially low-pass filter to filter the frames of video and downsample them. Therefore, we compute a full Laplacian pyramid for every frame of video [10].

After that we complete decomposing the frame of video to Laplacian pyramid, we then process temporal filtering on each spatial band. To extract the frequency band of interest, we apply a bandpass filter to every spatial frequency band. For example, while the blood is flowing through the face, the red color is changing that the variation we cannot see with our naked eyes. Because the color has passbands with sharp cut-off frequencies, we apply an ideal filter to extract the redness. However, low-order IIR filters may be more useful for motion magnification. Whether color amplification or motion magnification, we process temporal filtering using uniform filter for all spatial levels, and for all pixels within each level.

We have got the temporal filtered image, then our goal is to amplify the extracted signal by a magnification factor α that will improve the visual effects.

A. First-order Motion

Eulerian Video Magnification amplifies the pixel variation over the time at the Eulerian perspective. We then study the simple case of 1-D first-order Taylor serial expansion to realize the video magnification.

$I(x, t)$ denotes the image intensity at position x at time t . The initial state is expressed in (1).

$$I(x, 0) = f(x) \quad (1)$$

Since the image undergoes translations motion, the pixel value changes over the time. If the displacement function is represented as $\delta(t)$, we express the observed intensities at time t as following expression:

$$I(x, t) = f(x + \delta(t)) \quad (2)$$

Assuming the image can be approximated by a first-order Taylor serial expansion, so we get

$$I(x, t) = f(x) + \delta(t) \frac{\partial f(x)}{\partial x} \quad (3)$$

Then we apply temporal bandpass filter to $I(x, t)$, and assume that we extract the signal expect $f(x)$ in (3). We use $B(x, t)$ to represent the result. So we have

$$B(x, t) = \delta(t) \frac{\partial f(x)}{\partial x} \quad (4)$$

We have got signal of interest, then we amplify it by factor α and add it back to $I(x, t)$, the result is

$$\tilde{I}(x, t) = I(x, t) + \alpha B(x, t) \quad (5)$$

Combine (1), (2) and (3), we get

$$\tilde{I}(x, t) = f(x) + (1 + \alpha) \delta(t) \frac{\partial f(x)}{\partial x} \quad (6)$$

Finally we use the properties of first-order Taylor serial expansion, the amplified result is expressed as follow

$$I(x, t) = f(x + (1 + \alpha) \delta(t)) \quad (7)$$

This process shows that the image intensity $f(x)$ at time t has changed from $\delta(t)$ to $(1 + \alpha) \delta(t)$. The whole process is illustrated for a single sinusoid in Figure 2.

In the above analysis, we assume that the video need to be amplified within the bandpass of the temporal filter and without attenuation through the temporal filter. However, in general case, the temporal filter is not an ideal filter and the signal will be attenuated through it. We use λ_k to represent the attenuation factor. Then we get the bandpassed result through temporal filter.

$$B(x, t) = \sum_k \lambda_k \delta_k(t) \frac{\partial f(x)}{\partial x} \quad (8)$$

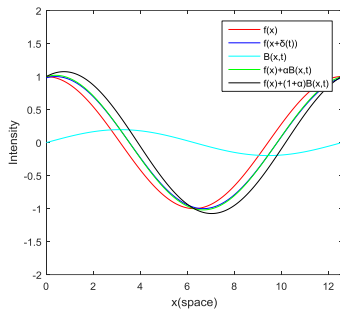


Figure 2. Motion magnification for a single sinusoid

We add the magnified signal back to the original signal. Comparison with (5), we will find that the attenuation of bandpass of the temporal filter can be interpreted as motion magnification factor $\alpha_k = \lambda_k \alpha$, so we get the motion magnified output

$$I(x,t) = f(x) + \sum_k (1 + \alpha_k) \delta_k(t) \quad (9)$$

B. Magnification Factor

We amplify the small motion using the first-order Taylor serial expansion, thus it's adapt to smooth image and small motion. For quickly changing image over time, the method becomes distorted. From (6), we know that the result will increase with larger magnification factor, larger displacement function and higher frequency. So the magnification factor α must be limited.

From above analysis, with the Taylor serial approximation, we get the processed signal is approximation equal to the true magnified output.

$$\begin{aligned} \tilde{I}(x,t) &\approx \hat{I}(x,t) \Rightarrow \\ f(x) + (1 + \alpha) \delta(t) \frac{\partial f(x)}{\partial x} &\approx f(x + (1 + \alpha) \delta(t)) \end{aligned} \quad (10)$$

Assuming $f(x) = \cos(\omega x)$, and $\beta = 1 + \alpha$, we get

$$\cos(\omega x) - \beta \omega \delta(t) \sin(\omega x) \approx \cos(\omega x + \beta \omega \delta(t)) \quad (11)$$

Using the cosines addition formula, we have

$$\begin{aligned} \cos(\omega x) - \beta \omega \delta(t) \sin(\omega x) &\approx \\ \cos(\omega x) \cos(\beta \omega \delta(t)) - \sin(\omega x) \sin(\beta \omega \delta(t)) \end{aligned} \quad (12)$$

In order to make sure (12) left is approximates equal to right, we must hold

$$\cos(\beta \omega \delta(t)) \approx 1 \quad (13)$$

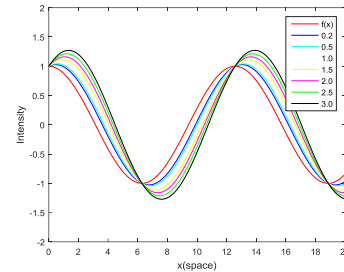
$$\sin(\beta \omega \delta(t)) \approx \beta \omega \delta(t) \quad (14)$$

According to (14), $\beta \omega \delta(t)$ need to meet the condition

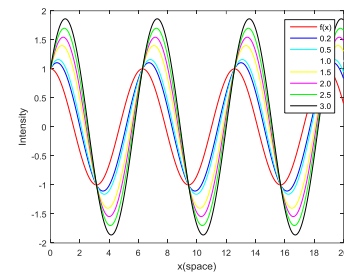
$$\beta \omega \delta(t) \leq \frac{\pi}{4} \quad (15)$$

With the knowledge of the spatial wavelength $\lambda = 4/\pi$, so the limitation is

$$(1 + \alpha) \delta(t) < \frac{\lambda}{8} \quad (16)$$



a) $\lambda = 4\pi$ and $\delta(1) = \pi/8$



b) $\lambda = 2\pi$ and $\delta(1) = \pi/8$

Figure 3. Motion magnification result for different spatial frequency

(16) shows that the magnification factor α must meet conditions when we amplify the small displacement $\delta(t)$ and image spatial wavelength λ . In theory, when we reveal the subtle variations, the magnification factor α is closer to the maximum, the magnified result is more obvious. However, in some special cases, the result will be more obvious when the magnification factor α is beyond the limitation. In those cases, the actual effect must be tested by experiments. Figure 3 shows the magnified result with different spatial wavelength and different magnification factor. The spatial wavelength will affect amplification. At the same magnification factor α , the spatial wavelength is longer, the magnified amplitude is smaller. So when we amplify the filtered Laplacian pyramid, we use different magnification factor α for different level of pyramid.

III. HARDWARE ARCHITECTURE OF EULERIAN VIDEO MAGNIFICATION ALGORITHM

Eulerian Video Magnification reveals the subtle variations combination that we cannot see with our naked eyes with spatial and temporal processing of video. When we amplify subtle temporal variation by Eulerian Video Magnification approach, we first decompose the video into different spatial frequency bands, then perform temporal filter to all bands to extract the interesting signal. The filter bands are multiplied

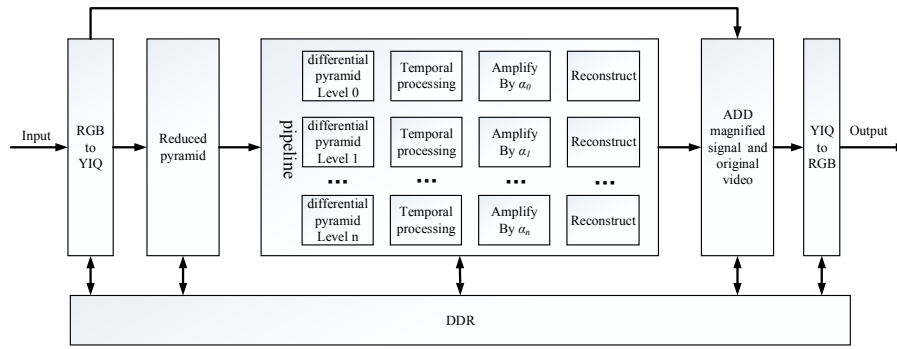


Figure 4. Hardware implementation of Eulerian Video Magnification

by magnification factor α . Finally, we add the magnified signal back to original to get the visible output. Nearly all Eulerian Video Magnification processes are compute-intensive calculation but low computational complexity which only include multiply and addition operations. This make it very suitable to implement in FPGA that has technologies of parallel computation and pipeline structure.

A. FPGA Implementation

According to above discussion, we design a hardware system to accelerate the Eulerian Video Magnification algorithm. The hardware implementation is illustrated in Figure 4. Firstly we convert the image from RGB space to YIQ space. Then we compute a reduced pyramid with n levels. For each level of reduced pyramid, the follow-up operations can be operated in pipeline that include differential pyramid, temporal processing, amplification and reconstruction. Finally, we add the magnified signal back to the original video and convert it to RGB space to get the visible output.

B. Laplacian Pyramid Implementation

The implementation of Laplacian pyramid consists of three sequential steps. It is illustrated in Figure 5. M and N denote the image size.

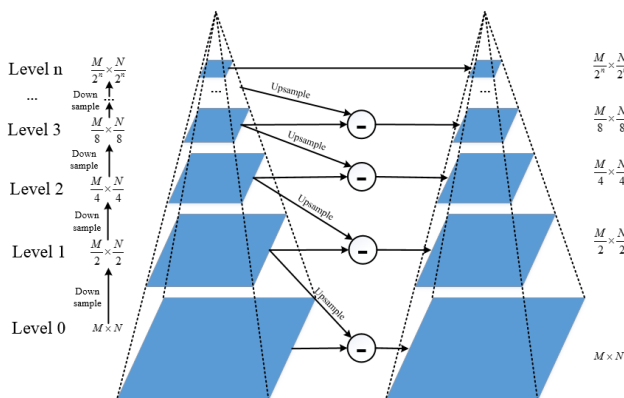


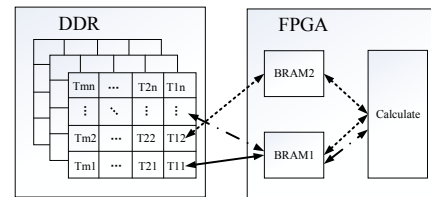
Figure 5. Algorithm of image Laplacian pyramid

- Compute the reduced resolution of the input image. First, the input image is filtered by a filter, such as binomial filter and Gaussian filter. Then we downsample the filtered result by extracting even lines and discarding odd lines. Finally, we repeat above

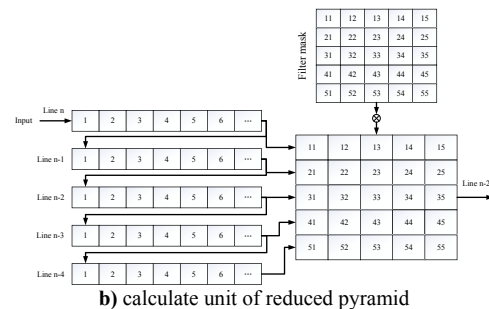
operations until we get the reduced pyramid with desired number of layers [12].

- Upsample the output of the previous step-again by a factor 2, and then filter the result. This creates a prediction image with the same resolution as the input.
- Compute the difference between the results of above two steps that is the result of Laplacian pyramid. This pyramid can be reconstructed progressively the original image, without distortion [12].

The whole image is too big to be stored in BRAM on FPGA. For example, if the image resolution is 1920×1080 , the whole image is 47.46Mb. In addition, because the reduced operation of is Laplacian pyramid done first. Therefore, difference operation, filter operation, amplification operation and reconstruction operation can be in parallel. However, BRAM source of XC7K325T-2FFG900C FPGA is 14Mb. In the paper, we divide the whole image into several tiles with the same size of 400×300 pixels. The memory access of reduced pyramid operation is illustrated in Figure 6(a). First, we read a tile from external DDR to FPGA BRAM1. Then FPGA calculates the reduced pyramid of this tile. As the same time, the next tile is transformed to FPGA BRAM2. After finishing computing data in BRAM1, FPGA starts calculating the BRAM2. As the same time, FPGA reads the third tile from DDR to BRAM1.



a) The memory access of reduced pyramid operation



b) calculate unit of reduced pyramid

Figure 6. FPGA implementation of reduced pyramid

After the data has been read to the BRAM of FPGA, then we compute the reduced pyramid. We use five shift registers to store the data to be processed. Five new pixels is inputted to shift register 1 in every clk. The last data of every shift registers is convoluted with filter mask. At the same time, the data in shift register is shifted to the right and the last data is moved to the first of next shift register. The five final results that we get are store as the line $n-2$. The calculate unit of reduced pyramid is illustrated in Figure 6(b).

C. Temporal Processing

After Laplacian pyramid has been finished, we apply the temporal filter specified by the user to extract the signal of interest. The passband frequencies and cut-off frequencies are critical for Eulerian Video Magnification. For example, the redness of face changes with the blood flows face. It results from the contractions of the heart. The normal adult human heart rates ranges from 40-100 bpm. We choose an ideal filter with sharp cut-off frequencies and a passband of 0.67 HZ to 1.67 HZ.

For ideal filter, we transform the signal to be filtered to frequency domain by Fourier Transform. Then we set the signal in passband to original value, but zero the signal in stopband. Finally, we transform the filter frequency signal to spatial domain by inverse Fourier Transform. We implement Fourier transform and inverse Fourier transform with Xilinx IP. The above temporal process indicates that the filtering operation can be processed sequentially, and pyramid levels can be processed simultaneously. We divide the temporal operation into 4-stage pipeline. It is illustrated in Figure 7.

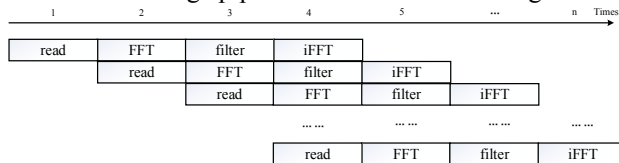


Figure 7. Four-stage pipeline of temporal processing

D. Memory Access

After we get the reduced pyramid, we do the follow-up operation. Because of differential pyramid, temporal processing and pyramid reconstruction, the memory must store two complete pyramids data at least. In order to save memory overhead, we divide BRAM into several parts that store the different data respectively. As above description of pipeline implementation, the memory access is illustrated in Figure 8. Firstly, we divide the reduced pyramid into several tiles with the same size 400×300 . Secondly, we read a tile of n level of i reduced pyramid and the corresponding tile of $n-1$ level of i reduced pyramid to BRAM in FPGA. FPGA has stored 2 tiles of a reduced pyramid. Then, we can do the differential pyramid. At the same time, FPGA reads the corresponding tile of n level of $i-1$ reduced pyramid to BRAM. The next step, we do the temporal filter and amplification operations. Simultaneously, FPGA reads the corresponding tile of n level of $i-1$ differential pyramid to BRAM. Finally, we reconstruct the pyramid and write the result to DDR. Using above method of memory access, the usage of BRAM

resource is about 15%, so the three channel of image can be processed in parallel.

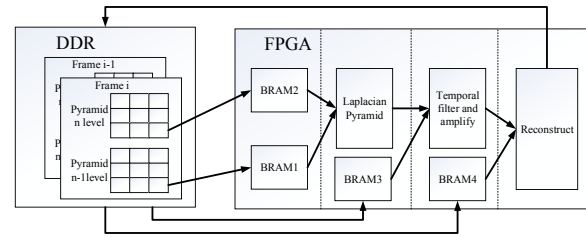


Figure 8. Memory access

IV. HARDWARE ARCHITECTURE OF EULERIAN VIDEO MAGNIFICATION ALGORITHM

As discussed above, we implement our design on Xilinx Kintex-7 KC705 board that contains a XC7K325T-2FFG900C FPGA chip and DDR3 with 4GB capacity.

A. Color Changes Amplification

In the paper, we firstly amplify the subtle color variations in face using the video with different resolutions. When we magnify these videos by Eulerian Video Magnification, we process the algorithm in YIQ space. A separable binomial filter of size five is selected to decompose the video to Laplacian pyramid. The value of magnification factor α is approximately set to 100, and the value of spatial wavelength λ_c is set to 1000. The cut-off frequencies ω_l and ω_h are set to 0.67 HZ and 1.67 HZ respectively. We test the face video on our FPGA board and compare the result with implementation on Intel i5 CPU 2.4GHZ with 8GB DDR3. The results are shown in Table 1. The processing time of CPU increases significantly with resolution. However, the processing time of FPGA increases little. In comparison with CPU, our hardware acceleration method significantly improves performance.

TABLE 1. PERFORMANCE OF CPU AND FPGA

Video Resolutions	Video Time	Performance		
		CPU	FPGA	Speedup
592×528[4]	10.00s	64.00s	7.03s	9.10x
1280×720	9.68s	177.09	8.95s	22.16x
1920×1080	10.35s	422.49s	10.84s	38.97x

We compare the visual effect using the video given by [4] with resolution 592×528 . Fig.9 (a) is the frame of the original video sequence, Fig. 9 (b) is same frames amplified by FPGA and Fig. 9 (c) is the amplified result of CPU. Obviously, the visual effect of result on FPGA is the same as CPU. However, the time on FPGA is nearly 9.10x speedup versus Eulerian Video Magnification on CPU.

B. Subtle Motion Magnification

We also test motion magnification using the “baby.mp4” video given by [4] that resolution is 960×544 , the time is 10.03 seconds and the frame rate is 30fps. We amplify the subtle motion around the chest when the baby is breathing. The value of magnification factor α is approximately set to 10. Baby’s chest motion magnification choose two first-order

lowpass IIR filter with cut-off frequencies $\omega_l=0.4$ Hz and $\omega_h=3$ Hz respectively. The temporal filter is different from ideal filter of color amplify. The time of implementation on Intel i5 CPU 2.4GHZ with 8GB DDR3 is 162.90s and the time on our FPGA board is 8.36s. By comparing the two result, we find that the FPGA is 19.48x speedup versus CPU and the magnified output video by FPGA is also nearly the same as CPU output.



a) Original frames



b) Output of magnified frames by FPGA



c) Output of magnified frames by CPU

Figure 9. Original and magnified frames

V. CONCLUSIONS AND FUTURE WORK

The paper discusses the Eulerian Video magnification algorithm and designs a hardware system that implements on Xilinx Kintex-7 FPGA with 4GB DDR3. Eulerian Video Magnification algorithm amplifies the video that combines the spatial and temporal processing but doesn't track or estimate the motion like Lagrangian method. It is suitable to reveal the subtle variation to be invisible with our naked eyes. Eulerian Video Magnification implementation mainly contains 4 parts: decomposing the video into Laplacian pyramid, performing temporal processing, amplifying the filtered video and reconstructing the video to be visible. Almost all Eulerian Video Magnification calculation only contains multiply and addition operation. It is very suitable to be processed by FPGA that has the advantages of parallel and pipeline architecture. Experiments shows that the output of magnified video by FPGA is basically the same as the output on CPU. However, the FPGA is running at 120 MHz, Intel i5 CPU with 2 cores is running at 2.4 GHz.

In the future, we will optimize Eulerian Video Magnification algorithm and hardware implementation. So that the system can amplify the video in real-time for 1080p videos and higher resolution videos.

ACKNOWLEDGMENT

We are very grateful for the video magnification group of MIT and the staffs' help of our laboratory.

REFERENCES

- [1] Poh, Ming-Zher, Daniel J. McDuff, and Rosalind W. Picard. "Non-contact, automated cardiac pulse measurements using video imaging and blind source separation." *Optics express* 18.10 (2010): 10762-10774.
- [2] Liu, Ce, et al. "Motion magnification." *ACM transactions on graphics (TOG)* 24.3 (2005): 519-526.
- [3] Verkrusse, Wim, Lars O. Svaasand, and J. Stuart Nelson. "Remote plethysmographic imaging using ambient light." *Optics express* 16.26 (2008): 21434-21445.
- [4] Wu H Y, Rubinstein M, Shih E, et al. "Eulerian video magnification for revealing subtle changes in the world". *ACM transactions on graphics (TOG)* 31.4 (2012): 13-15.
- [5] Amara, Amara, Frédéric Amiel, and Thomas Ea. "FPGA vs. ASIC for low power applications." *Microelectronics Journal* 37.8 (2006): 669-677.
- [6] Asano, Shuichi, Tsutomu Maruyama, and Yoshiki Yamaguchi. "Performance comparison of FPGA, GPU and CPU in image processing." *2009 international conference on field programmable logic and applications*. IEEE, 2009.
- [7] Pauwels, Karl, et al. "A comparison of FPGA and GPU for real-time phase-based optical flow, stereo, and local image features." *IEEE Transactions on Computers* 61.7 (2012): 999-1012.
- [8] Lucas, Bruce D., and Takeo Kanade. "An iterative image registration technique with an application to stereo vision." *IJCAI*. Vol. 81. No. 1. 1981.
- [9] Horn, Berthold KP, and Brian G. Schunck. "Determining optical flow." *Artificial intelligence* 17.1-3 (1981): 185-203.
- [10] Burt, Peter, and Edward Adelson. "The Laplacian pyramid as a compact image code." *IEEE Transactions on communications* 31.4 (1983): 532-540.
- [11] Ahirwal, Balkrishan, Mahesh Khadare, and Rakesh Mehta. "FPGA based system for color space transformation RGB to YIQ and YCbCr." *Intelligent and Advanced Systems, 2007. ICIAS 2007. International Conference on*. IEEE, 2007.
- [12] Gonzalez, Rafael C., R. E. Woods, and B. R. Masters. "Digital Image Processing, Third Edition." *Journal of Biomedical Optics* 14.2(2009):331 - 333.
- [13] Song, Yajun, et al. "Implementation of real-time Laplacian pyramid image fusion processing based on FPGA." *Photonics Asia 2007. International Society for Optics and Photonics, 2007*.



Kening Zhang received his B.S. degree in 2014 from School of Chang'an University, Shaanxi Province, China, and he is currently a M.S. student in Department of Physics in University of Science and Technology of China, Anhui, China, under the supervision of Prof. Xi Jin. His current research work is mainly on FPGA-based Hardware Accelerator design and image processing.



Xi Jin received the B.S. degree from University of Science and Technology of China, Anhui, China, and he is currently an associate professor in Department of Physics in University of Science and Technology of China, Anhui, China. His research interests include SOC design technology, VLSI design, computer-aided design methodologies for SoC system integration and FPGA-based Hardware structure design.



An Wu received his B.S. degree in 2011 from School of Anhui University, Anhui province, China, and he is currently a Ph.D. student in Department of Physics in University of Science and Technology of China, Anhui, China, under the supervision of Prof. Xi Jin. His current research work is mainly on SoC design technology, VLSI design and FPGA-based Hardware Accelerator design.