

Chapter 3

Real-time Capillary-level Microchannel Flow Analysis Using a Full-pixel Frame-straddling Micro-PIV System

Idaku Ishii and Tadayoshi Aoyama

Abstract We develop a high throughput microscopic particle image velocimetry system that can compute flow vectors of 512×512 pixels in real time at 500 fps for fast microchannel flow. To compute many flow vectors at high speed, a gradient-based optical flow method is accelerated by calculating integral images of product sums of image brightness gradients and implementing them as parallel processes on a GPU-based frame-straddling high-frame-rate (HFR) vision system. Thus, the HFR vision system, having two cameras with a time delay function, can simultaneously compute hundreds of millions of flow vectors in a second, assuming a small image displacement between frames with a submillisecond delay. We conducted real-time flow measurement experiments to quantify capillary-level microchannel flow in a microfluidic chip with many 7- μm -width channels, and verified the high throughput performance of our system for long-term microchannel blood flow analysis.

Keywords Microscopic flow measurement · Particle image velocimetry (PIV) · Real-time image processing

3.1 Introduction

Microscopic particle image velocimetry (micro-PIV) [1, 2] is a well-known method of flow visualization that obtains the velocities and related properties of microchannel flows from videos. Micro-PIV systems have been widely used in various applications such as pressure-driven microchannel flow [3], laminar flow and turbulence transition analysis [4], 3-D flow analysis based on stereo principles [5], and inkjet printhead analysis [6]. Such micro-PIV techniques enable microchannel flow measurement with high space resolution. Further, many studies on blood flow have been reported for understanding blood flow behavior in microcirculation:

I. Ishii (✉) · T. Aoyama
Department of System Cybernetics, Hiroshima University,
1-4-1, Kagamiyama, Higashi-Hiroshima, Japan
e-mail: iishii@robotics.hiroshima-u.ac.jp

Sugii et al. [7] reported that blood flow in a straight microchannel had a blunt velocity profile, Chiu et al. [8] analyzed the effect of blood flow on monocyte adhesion to endothelial cells, Kim et al. [9] analyzed the blood behavior in a circular opaque microchannel using an X-ray PIV technique, Venneman et al. [10] measured the blood-plasma velocity in the beating heart of a chicken embryo, and Lima et al. [11] analyzed in vitro blood flow behavior with submicrometer optical thickness using a confocal micro-PIV system. Most these micro-PIV studies processed high-frame-rate (HFR) videos for the apparent fast microchannel flows, because the apparent velocity of microchannel flow in a microscopic image becomes larger as magnification is increased.

However, most micro-PIV systems processed the HFR videos offline for post microchannel flow analysis and human visualization because flow in these systems was estimated by cross-correlation based methods [12, 13], which require heavy computation for flow estimation. HFR video-based systems were limited to flow analysis for short time spans because offline HFR cameras had insufficient memory. Thus, current micro-PIV technology cannot always function as a real-time sensor to measure microchannel flow for long observation periods. To realize real-time micro-PIV, we developed a real-time frame-straddling HFR vision system [14] that can synchronize two camera inputs for the same camera view with a time delay on the order of microseconds. Assuming a small image displacement between frames with a tiny time delay, fast microchannel flow was simultaneously estimated using a gradient-based optical flow method, which is more suitable for real-time processing than cross-correlation-based PIV methods because flow distribution is estimated by calculating local brightness gradients, which do not need heavy computation. However, the number of measurement points per unit of time is limited by the performance of the personal computer (PC) in [14] because the optical flow estimation was executed in software on a PC. If we could accelerate the optical flow estimation to increase the number of measurement points in micro-PIV, full-pixel microchannel flow sensing could be conducted in real time at a high frame rate, enabling long-term microchannel flow analysis with high space and time resolution without the need for flow sensors to be physically attached.

This study develops a high-throughput micro-PIV system that can simultaneously compute hundreds of millions of flow vectors in a second for fast microchannel flow by accelerating a gradient-based optical flow algorithm on a GPU-based frame-straddling HFR vision system, which has two cameras with a submillisecond-delay function. The optical flow estimation was accelerated by computing integral images for product sums of brightness gradients of input images, which are required in the gradient-based optical flow method, and implementing them as parallel processes on the GPU-based frame-straddling HFR vision system.

Full-pixel flow vectors of 512×512 pixels can be computed for fast microchannel flow in real time at 500 fps. The performance of our system is verified using experimental results to quantify the spatio-temporal transitions of horse blood fast-flowing in a 7- μm -width microchannel array at the capillary level.

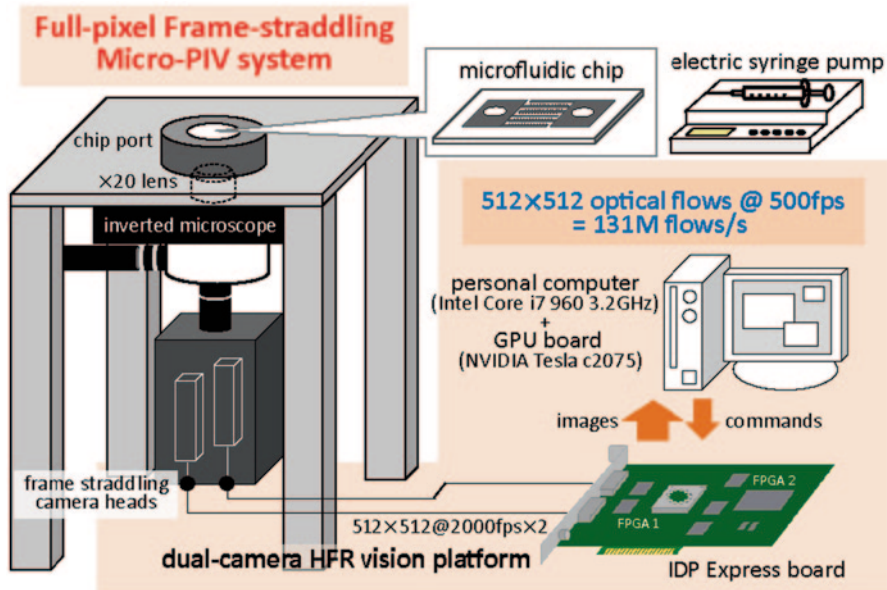


Fig. 3.1 System configuration

3.2 Frame-Straddling Micro-PIV System

Our micro-PIV system, an improvement on our previously developed frame-straddling micro-PIV system [14], comprises an HFR vision platform (IDP Express) [15] with a frame-straddling function for two camera inputs, a GPU board (Tesla c2075; NVIDIA Co., USA), a personal computer (PC), an inverted microscope with a microfluidic chip with $7\text{-}\mu\text{m}$ wide channels (MWA-MCFANbasic; Kikuchi Microtechnology Co., Japan), and an electric syringe pump (KDS200; KD Scientific Inc., USA). Figures 3.1 and 3.2 show its configuration and overview, respectively.

The IDP Express was designed to implement real-time video processing and recording of 512×512 images at 2000 fps. It consists of two camera heads and a dedicated FPGA image processing board (IDP Express board). The camera head has a CMOS image sensor of 512×512 pixels; its sensor and pixel sizes are $5.12 \times 5.12 \text{ mm}$ and $10 \times 10 \mu\text{m}$, respectively.

It can capture 8-bit gray-level 512×512 images at 2000 fps. The dimensions and weight of the camera head are $23 \times 23 \times 77 \text{ mm}$ and 145 g. Figure 3.2b shows a camera housing mounted at the camera port of the microscope in which two camera heads of the IDP Express are arranged so that they can capture a common view via a prism. The IDP Express board has two camera inputs, so that two 512×512 images and their processed results can be mapped onto the standard memory of a PC at 2000 fps via a PCI-e bus. In this study, the IDP Express board was dedicated for dual-camera frame-straddling by improving the hardware logic for time delay

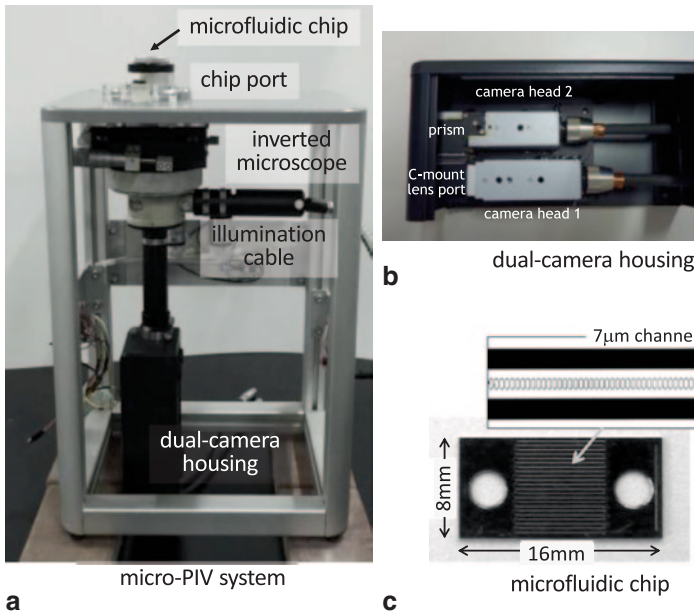


Fig. 3.2 System overview

control between the two camera heads; the image capture time delay is controlled via the PC from 0 to 0.5 ms in 9.9 ns steps.

The Tesla c2075 is a computing processor board accelerated by a NVIDIA Fermi GPU GF110. It has a processing performance of 1.03 TFlops using 448~processor cores operating at 1.15 GHz, a bandwidth of 144 GB/s, an inner global memory of 6 GB, and fast shared memory of 64 kB. A PC with 16-lane PCI-e 2.0 buses and a processor chipset with DMA were adopted to transfer memory-mapped data between standard memory and the Tesla c2075 via the PCI-e bus. We used a PC with an ASUS P6T7 WS SuperComputer motherboard, Intel Core™ i7 960 3.2 GHz CPU, and 3 GB RAM. We used a CUDA IDE provided by NVIDIA for coding the algorithms that had dedicated API functions for IDP Express in Windows 7 (32 bit), enabling us to access memory mapped data.

In this study, we observed microchannel flows in many 7- μm -width channels in an 8×16 mm-size microfluidic chip device of 0.5 mm thickness [16], as shown in Fig. 3.2c. It was fabricated of single crystal silicon for flow assessment at the capillary level. The microfluidic chip had 7854 channels, and the width, length, and depth of the channel were 7, 30, and 4.5 μm , respectively. Thirteen channels of them were observed under a 20X objective lens on the microscope, illuminated in the microscopic view by a metal-halide light source (PCS-MH375RC; Optical Garden Co., Japan). The measurement area was $256 \times 256 \mu\text{m}$, and one pixel corresponded to 0.5 μm in the 512×512 pixels microscopic view on the 10 μm -pixel-pitch image sensor.

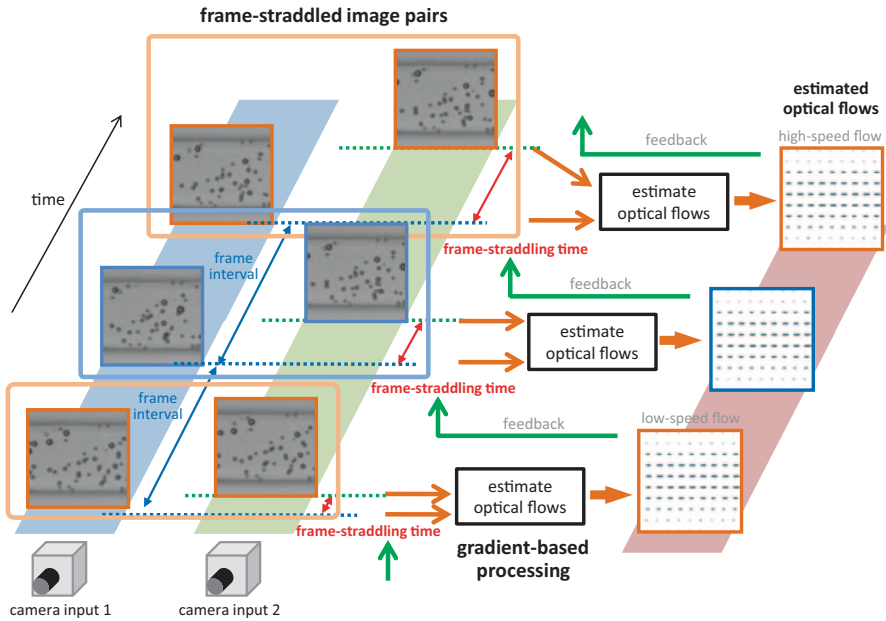


Fig. 3.3 Frame-straddling optical flow concept

3.3 Frame-Straddling Optical Flow Estimation Using Integral Gradient Images

3.3.1 Frame-Straddling Optical Flow

We proposed a VFS-OF algorithm [14] that improves on the Lucas-Kanade method [17] and is designed to estimate optical flow accurately for both high- and low-speed flows in the same system.

Figure 3.3 shows the VFS-OF method. Given a dual-camera high-speed vision system that synchronizes two camera inputs for the same view field with a time delay, the VFS-OF method adjusts its time delay for accurate, real-time gradient-based flow estimation so image displacement between frame-straddled images is controlled optimally to be small—around one pixel per frame on the subpixel order.

Compared to single-camera optical flow estimation at a fixed frame rate, VFS-OF improves the measurable velocity range by adjusting frame-straddling time to the amplitude of measured flow vectors. The VFS-OF method thus simultaneously estimates microchannel flow vectors at a variety of speeds while microchannel flow velocity is fluctuating rapidly in time and undergoing large changes in magnitude.

Here we assume that the two camera input images of a dual-camera high-speed vision system, ${}^1I(x, y, t)$ (camera input (1) and ${}^2I(x, y, t)$ (camera input 2), match perfectly:

$${}^1I(x, y, t) = {}^2I(x, y, t). \quad (3.1)$$

Product sums S_{xx} , S_{xy} and S_{yy} at t related to partial derivatives of the space direction use image of ${}^1I(x, y, t)$ (camera input 1) the same as for the Lucas-Kanade method. When the dual-camera frame interval is Δt , product sums related to partial derivatives of the time direction are calculated by dual-camera input ${}^1I(x, y, t)$, ${}^2I(x, y, t + \tau(t - \Delta t))$ with time delay $\tau(t - \Delta t)$ determined by $t - \Delta t$.

$$\begin{aligned} S_{xx} &= \sum_{\Gamma(x,y)} {}^1I_x {}^1I_x, \\ S_{xy} &= \sum_{\Gamma(x,y)} {}^1I_x {}^1I_y, \\ S_{yy} &= \sum_{\Gamma(x,y)} {}^1I_y {}^1I_y, \\ S'_{xt} &= \sum_{\Gamma(x,y)} {}^1I_x \frac{{}^1I(x, y, t) - {}^2I(x, y, t - \tau(t - \Delta t))}{\tau(t - \Delta t)}, \\ S'_{yt} &= \sum_{\Gamma(x,y)} {}^1I_y \frac{{}^1I(x, y, t) - {}^2I(x, y, t - \tau(t - \Delta t))}{\tau(t - \Delta t)}. \end{aligned} \quad (3.2)$$

Partial derivatives of space direction 1I_x , 1I_y related to ${}^1I(x, y, t)$ are calculated as follows:

$$\begin{aligned} {}^1I_x &= \frac{1}{2} ({}^1I(x+1, y, t) - {}^1I(x-1, y, t)), \\ {}^1I_y &= \frac{1}{2} ({}^1I(x, y+1, t) - {}^1I(x, y-1, t)). \end{aligned} \quad (3.3)$$

Frame-straddling time between the two camera inputs $\tau(t - \Delta t)$ is adaptively adjusted using flow vectors ($v_x(x, y, t - k\Delta t)$, $v_y(x, y, t - k\Delta t)$) measured at time $t - k\Delta t$ ($k = 1, \dots, K$) calculated for K frames. This time-averaged flow vector distribution is calculated as follows:

$$\begin{aligned} &(\tilde{v}_x(x, y, t - \Delta t), \tilde{v}_y(x, y, t - \Delta t)) \\ &= \left(\frac{1}{K} \sum_{k=1}^K v_x(x, y, t - k\Delta t), \frac{1}{K} \sum_{k=1}^K v_y(x, y, t - k\Delta t) \right). \end{aligned} \quad (3.4)$$

The time delay for a dual camera is determined by $(\tilde{v}_x(x, y, t - k\Delta t), \tilde{v}_y(x, y, t - k\Delta t))$ after calculating time-averaged flow vector distribution as follows:

$$\tau(t - \Delta t) = \frac{A}{\max_{x,y} \left(\sqrt{\tilde{v}_x^2(x, y, t - \Delta t) + \tilde{v}_y^2(x, y, t - \Delta t)} \right)}, \quad (3.5)$$

where A is a constant that indicates the control target to estimate flow velocity. A is perfectly set to a small value of around one pixel or less on the subpixel order in the case of optical flow estimation using the Lucas-Kanade method.

Based on the amplitudes of measured flow vectors, frame-straddling time $\tau(t - \Delta t)$ is adjusted in Eq. (3.5), a small frame-straddling time is set for high-speed flow, and a large frame-straddling time is set for low-speed flow. Flow vector $(v_x(x, y, t), v_y(x, y, t))$ at time t is thus estimated by controlling frame-straddling time based on flow vectors $(v_x(x, y, t - k\Delta t), v_y(x, y, t - k\Delta t))$ ($k = 1, \dots, K$) measured by time $t - \Delta t$ using the following equation:

$$\begin{pmatrix} v_x \\ v_y \end{pmatrix} = \frac{1}{S_{xx}S_{yy} - S_{xy}^2} \begin{pmatrix} S_{yy}S'_{xt} - S_{xy}S'_{yt} \\ -S_{xy}S'_{xt} + S_{xx}S'_{yt} \end{pmatrix} \quad (3.6)$$

Using the frame-straddling optical flow method with a time delay on the order of microseconds, we measured fast microchannel flow accurately in real time [14], however, the number of measurement points and the output rate of flow vectors were limited by PC performance. Two limited modes were implemented when $\Gamma(x, y)$ was set to a neighbor region of 16×16 pixels ($m=16$): (1) intersection mode (16 points, 1000 fps), and (2) whole-image mode (16×16 points, 50 fps). This is because the frame-straddling optical flow method was implemented in software on a PC, and its computational complexity depended on the size of $\Gamma(x, y)$. The frame-straddling optical flow method in [14] requires a computational complexity of $O(fm^2N^2)$ when flow velocities for all pixels of an $N \times N$ image are estimated using an $m \times m$ cell to calculate the product sums of brightness gradients at frame rate f .

Thus, to realize full-pixel flow estimation for real-time microchannel flow analysis with high space and time resolution, it is necessary to reduce the computational complexity of the gradient-based optical flow method so that it is independent of the cell size used in the brightness gradient calculation. It is also important to increase the computational power available for optical flow estimation by a parallel implementation on a GPU-based HFR vision system.

3.3.2 Integral Gradient Images

In this study, the computational complexity needed to calculate the product sums of brightness gradients in Eq. (3.2) is reduced by calculating the following five

integral gradient images ii_{xx} , ii_{xy} , ii_{yy} , ii_{xt} , and ii_{yt} for the product sum of the brightness gradients, I_x^2 , $I_x I_y$, I_y^2 , $I_x I_t$ and $I_y I_t$, respectively, as follows:

$$s_{\xi\eta}(x, y) = s_{\xi\eta}(x-1, y) + M(x, y) I_{\xi}(x, y, t) I_{\eta}(x, y, t), \quad (3.7)$$

$$ii_{\xi\eta}(x, y) = ii_{\xi\eta}(x, y-1) + s_{\xi\eta}(x, y) \quad (\xi, \eta = x, y, t), \quad (3.8)$$

where $s_{\xi\eta}(x, y)$ is the cumulative row sum, the boundary conditions are $s_{\xi\eta}(-1, y) = 0$ and $ii_{\xi\eta}(x, -1) = 0$, and $M(x, y)$ is a one-bit mask image consisting of preassigned observable regions. The integral image of a gray-level image has been used to accelerate the computation of Haar-like features for fast face detection in [18], and our integral gradient image is its expansion for brightness gradient images.

When $\Gamma(x, y)$ is a region of $m \times m$ pixels, whose starting point is (a, b) , the product sums of brightness gradients, $S_{\xi\eta}$, can be computed using the values of integral gradient images on the four vertices of $\Gamma(x, y)$ as follows:

$$S_{\xi\eta} = ii_{\xi\eta}(a, b) + ii_{\xi\eta}(a+m-1, b+m-1) - ii_{\xi\eta}(a+m-1, b) - ii_{\xi\eta}(a, b+m-1). \quad (3.9)$$

Using the integral gradient images, we can accelerate the computation of the product sums of brightness gradients, even for computations that uses a large cell-size because the computational complexity is independent of cell size. We can estimate flow vectors at all pixels of $N \times N$ images in the frame-straddling optical flow method with a computational complexity of $O(N^2)$, dependent only on image size.

3.3.3 Implemented Algorithm and Its Specifications

For full-pixel level optical flow estimation at high speed, we implemented the improved frame-straddling optical flow method as parallel logic on a GPU board in our dual-camera HFR vision system.

The algorithm implemented on the dual-camera HFR vision system has the following steps:

1. Image acquisition
Two 8-bit gray-level 512×512 input images, ${}^1B(x, y, t)$, and ${}^2B(x, y, t - \tau)$, captured from the two camera heads with time delay τ , are memory-mapped into PC memory.
2. Transfer of input images to GPU memory

The two PC-memory mapped images are transferred to the global memory on the GPU board.

3. Image correction

To align the locations and brightnesses of the two input images, the input image of camera 2 ${}^2B(x, y, t - \tau)$ is corrected based on that of camera 1 ${}^1I(x, y, t) = {}^1B(x, y, t)$ as follows:

$${}^2I(x, y, t) = \text{Blut}(\text{Warp}({}^2B(x', y', t))), \quad (3.10)$$

where $\text{Blut}(\cdot)$ is a lookup table that shows the relationship between the brightnesses of cameras 1 and 2. $\text{Warp}(\cdot)$ is a function to transform the non-integer coordinates of a pixel to integers after bilinear interpolation. The non-integer coordinates $(x', y') = (a_1x + a_2y + a_5, a_3x + a_4y + a_6)$ of camera 2 are given by integer coordinates (x, y) of camera 1 along with the affine parameters of the geometric relationship between the images of cameras 1 and 2.

The correction process expressed in (3.10) is conducted in parallel on the GPU board for 16×16 blocks of 32×32 pixels.

The brightness lookup table $\text{Blut}(\cdot)$ and the affine parameters $a_i (i = 1, \dots, 6)$ are calculated a priori.

4. Calculation of integral gradient images

Using the two corrected images, ${}^1I(x, y, t)$ and ${}^2I(x, y, t)$, five products of brightness gradients, I_x^2 , $I_x I_y$, I_y^2 , $I_x I_t$ and $I_y I_t$, are calculated in parallel on the GPU board for 16×16 blocks of 32×32 pixels.

Each of the five brightness gradient image $I_{\xi} I_{\eta}$ is scanned in the x direction to accumulated its cumulative row sum $s_{\xi\eta}(x, y)$ in parallel on the GPU board for 32×32 blocks of 16×16 pixels.

Next, each cumulative row sum $s_{\xi\eta}(x, y)$ is accumulated in parallel in the y direction to form its integral gradient image $ii_{\xi\eta}(x, y)$ on the GPU board for 32×32 blocks of 16×16 pixels.

5. Calculation of flow vector images

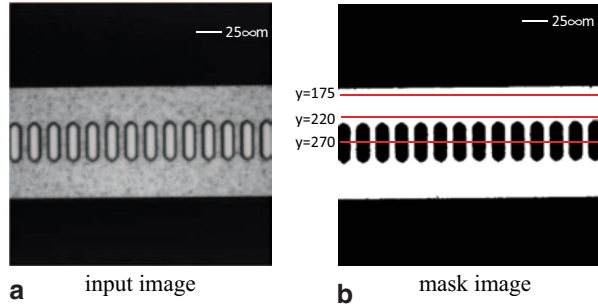
For each integral gradient image, the product sum of brightness gradient $S_{\xi\eta}$ is calculated by selecting the values of $ii_{\xi\eta}(x, y)$ on the four vertices of the $m \times m$ region $\Gamma(x, y)$, as expressed in Eq. (3.9). This process is conducted in parallel on the GPU board for 16×16 blocks of 32×32 pixels.

A full-pixel 512×512 flow vector image that gives the flow vectors (v_x, v_y) for all pixels of the input images is estimated using the five product sums of the brightness gradients, as expressed in Eq. (3.6). This process includes a time-averaging process using the flow vectors estimated at the ten previous frames for noise reduction; it is conducted in parallel on the GPU board for 16×16 blocks of 32×32 pixels.

Table 3.1 shows the execution times of the improved frame-straddling optical flow method when the flow vector images were calculated on our dual-camera HFR vision system; the execution times were independent of cell size when calculating the product sums of brightness gradients. The steps (2)-(5) were accelerated by executing them in parallel on the GPU board, and the total execution time for

Table 3.1 Execution times

Process	Time [ms]
Image acquisition	0.20
Image transfer CPU to GPU	0.18
Image correction	0.10
Integral gradient image calculation	0.72
Flow vector image calculation	0.40
Total	1.60

Fig. 3.4 Microchannels to be observed. **a** input image, **b** mask image

acquiring a flow vector image of 512×512 pixels was within 1.60 ms, including the data transfer time from the PC memory to the GPU board. We confirmed that the flow vectors at 512×512 points were estimated in real time at 500 fps; our improved micro-PIV system can compute 131 million flow vectors per second, eight thousand times more quickly than our previously reported real-time micro-PIV measurement [14].

3.4 Experiments

To verify the performance of our system, several experiments were conducted using defibrinated horse blood, which were flowing in a $7\text{-}\mu\text{m}$ -width microchannel array under a $20\times$ objective lens.

The liquids to be observed were supplied to the microfluidic chip using an electric syringe pump, and flowed through the channels from the lower to upper portions of the image.

Figure 3.4a shows the microscopic view. Microchannel flows through thirteen channels can be observed in the sufficiently illuminated region in the middle of the image. However, it was too dim to observe microchannel flows in the upper and lower regions in the image because of the channel structure of the microfluidic chip.

We preassigned a one-bit mask image $M(x, y)$ for the observable regions as shown in Fig. 3.4b, excluding the microchannel walls and dark areas.

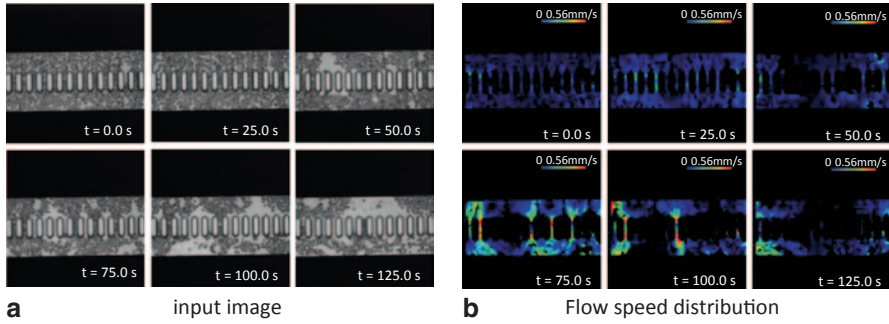


Fig. 3.5 Estimated microchannel blood flows. **a** input image, **b** Flow speed distribution

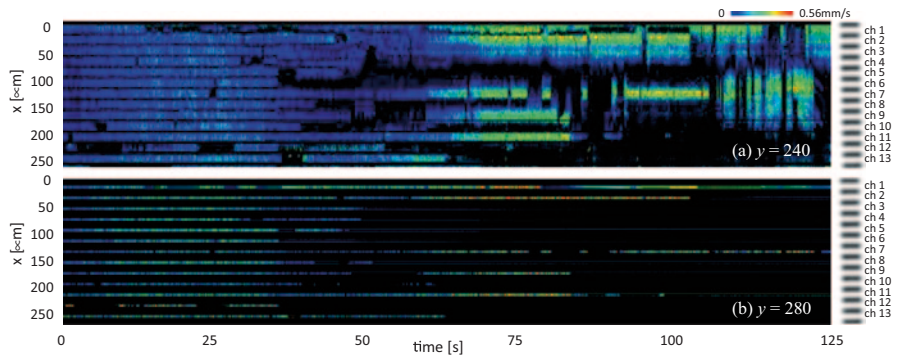


Fig. 3.6 Time-transient flow speeds for microchannel blood flow

The affine parameters and brightness lookup table between the two input images were calculated during prior dual-camera calibration. The exposure time of the camera heads was 50 μ s.

The cell size for the optical flow estimation was set to 32×32 pixels ($m = 32$) in the experiments.

Time-transient microchannel flow distributions were measured when defibrinated horse blood was supplied to the microfluidic chip. The defibrinated horse blood used in the experiment was provided by Kojin Bio Co., Japan. The blood cells, having a diameter of 5–6 μ m, were used as tracer particles for PIV measurements. The horse blood was supplied to the microfluidic chip at a flow rate of 25 μ l/min. The frame-straddling time was =600 μ s.

Figure 3.5 shows (a) a six image sequence of camera 1, and (b) the estimated flow speed distributions at intervals of 25 s. Here $t = 0$ was the start time of observation, and the duration of observation was 125 s. Figure 3.6 shows the time-transient profiles of flow speeds estimated on the intersection lines; $y = 240$ around the upper exits of the channels, and $y = 280$ at the centers of the channels. Figure 3.7 shows the temporal changes of flow speeds estimated at points at the centers of the channels.

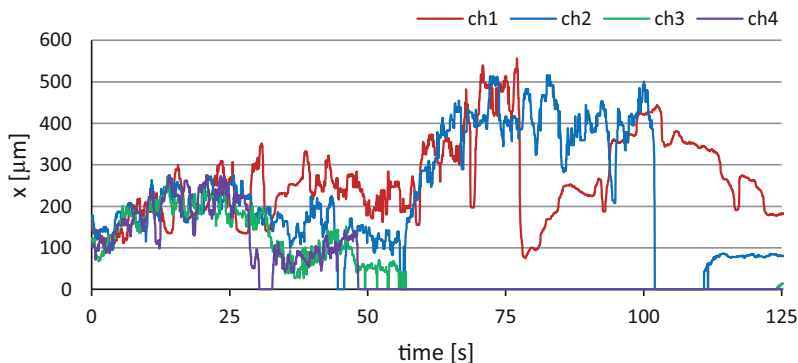


Fig. 3.7 Flow speeds at centers of microchannels $y=280$

In Fig. 3.7, the noise in the estimated flow speeds was reduced with median filter over sample data in a second.

It was observed that blood flowed in all the channels at the start of observation, however, the number of blocked channels increased as time passed; time-varying irregular flow paths were generated because of the blocked channels. The numbers of blocked or stagnant channels were 0, 1, 5, 8, 10 and 11 at $t=0, 25, 50, 75, 100$ and 125 s, respectively. This is because blood cells collected in the entrance of the channel and blocked it when they adhered to each other. In Fig. 3.7, the flow speeds estimated at channels 2, 3, and 4, ((62, 280), (100, 280), and (138, 280) in image (x, y) coordinates), in which blood flows were stopped in the middle of observation, are plotted, compared to those points estimated in channel 1, (27, 280), in which blood cells continuously flowed over 125 s. For channels 2, 3 and 4, the flow speeds estimated at $t=0$ s were 174, 115, and 112 μs , respectively, whereas blood flows were almost stopped at $t=102, 57,$ and 48 s, respectively. For channel 1, the flow speed estimated at $t=0$ s was 169 $\mu\text{m/s}$, whereas the flow speed of channel 1 increased and irregularly changed as time passed; the flow speeds estimated at $t=75$ and 125 s were 503 and 182 $\mu\text{m/s}$, respectively. This is because the increase of blocked channels affected the flow speeds in the remaining non-blocked channels where blood smoothly flowed, and the number of blocked channels did not increase monotonically; blocked or non-blocked states were frequently perturbed in several channels.

3.5 Conclusion

In this study, we developed a real-time micro-PIV system that can simultaneously compute 131 million flow vectors per second; flow vectors of 512×512 pixels were estimated for fast microchannel flow in real time at 500 fps. Spatio-temporal transitions of blood flowing in a capillary-level microchannel array are quantified to

verify the performance of our system for the long-term analysis of microchannel blood flow. We plan to develop an LOC-based long-term blood flow observation in a microchannel to monitor and quantify its spatio-temporal changing flow distribution, corresponding to the state of blood cells in capillary blood circulation.

References

1. Santiago JG, Wereley ST, Meinhart CD, Beebe DJ, Adrian RJ (1998) A particle image velocimetry system for microfluidics. *Exp Fluids* 25(4):316–319
2. Mielnik MM, Saetran LR (2005) Micro particle image velocimetry: an overview. *Exp Fluids* 38(3):1–8
3. Meinhart CD, Wereley ST, Santiago JG (1999) PIV measurements of a microchannel flow. *Exp Fluids* 27(5):414–419
4. Sharp KV, Adrian RJ (2004) Transition from laminar to turbulent flow in liquid filled microtubes. *Exp Fluids* 36(5):741–747
5. Klank H, Goranovic G, Kutter JP, Gjelstrup H, Michelsen J, Westergaard CH (2002) PIV measurements in a microfluidic 3D sheathing structure with three-dimensional flow behaviour. *J Micromech Microeng* 12(6):862–869
6. Meinhart CD, Zhang H (2000) The flow structure inside a microfabricated inkjet printhead. *J Microelectromech Syst* 9(1):67–75
7. Sugii Y, Nishio S, Okamoto K (2002) In vivo PIV measurement of red blood cell velocity field in microvessels considering mesentery motion. *Physiol Meas* 23(2):403–416
8. Chiu J, Chen C, Lee P, Yang C, Chuang H, Chien S, Usami S (2003) Analysis of the effect of distributed flow on monocytic adhesion to endothelial cells. *J Biomech* 36(12):1883–1895
9. Kim G, Lee S (2006) X-ray PIV measurements of blood flows without tracer particles. *Exp Fluids* 41(2):195–200
10. Vennemann P, Kiger KT, Lindken R, Groenendijk BCW, Stekelenburg-DeVos S, Ten Hagen TLM, Ursem NTC, Poelmann RE, Westerweel J, Hierck BP (2006) In vivo micro particle image velocimetry measurements of blood-plasma in the embryonic avian heart. *J Biomech* 39(7):1191–1200
11. Lima R, Wada S, Tanaka S, Takeda M, Ishikawa T, Tsubota K, Imai Y, Yamaguchi T (2008) In vitro blood flow in a rectangular PDMS microchannel: experimental observations using a confocal micro-PIV system. *Biomed Microdevices* 10(2):153–167
12. Hart DP (2000) Super-resolution PIV by recursive local-correlation. *J Visual* 3(2):13–22
13. Sugii Y, Nishio S, Okuno T, Okamoto K (2000) highly accurate iterative PIV technique using a gradient method. *Meas Sci Technol* 11(12):1666–1673
14. Kobatake M, Takaki T, Ishii I (2012) A real-time micro-PIV system using frame-straddling high-speed vision. *Proceedings of the IEEE International Conference Robotics Automation* pp 397–402, St. Paul, Minnesota, May 2012
15. Ishii I, Tatebe T, Gu Q, Moriue Y, Takaki T, Tajima K (2010) 2000 fps real-time vision system with high-frame-rate video recording. *Proceedings of IEEE International Conference on Robotics and Automation* pp 1536–1541, Anchorage, Alaska, May 2010
16. Kikuchi Y, Sato K, Mizuguchi Y (1994) Modified cell flow microchannels in a single-crystal silicon substrate and flow behavior of blood cells. *Microvasc Res* 47(1):126–139
17. Lucas B, Kanade T (1981) An iterative image registration technique with applications in stereo vision. *Proceedings of the DARPA Image Understanding Workshop* pp 121–130, Washington DC, April 1981
18. Viola P, Jones M (2001) Rapid object detection using a boosted cascade of simple features. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp 511–518, Kauai, Hawaii, December 2001