Digital signal processor-based real-time optical Doppler tomography system

Shikui Yan
Daqing Piao
Yueli Chen
Quing Zhu
University of Connecticut
Electrical & Computer Engineering Department
Storrs, Connecticut 06269-2157
E-mail: zhu@engr.uconn.edu

Abstract. We present a real-time data-processing and display unit based on a custom-designed digital signal processor (DSP) module for imaging tissue structure and Doppler blood flow. The DSP module is incorporated into a conventional optical coherence tomography system. We also demonstrate the flexibility of embedding advanced Doppler processing algorithms in the DSP module. Two advanced velocity estimation algorithms previously introduced by us are incorporated in this DSP module. Experiments on Intralipid flow demonstrate that a pulsatile flow of several hundred pulses per minute can be faithfully captured in M-scan mode by this DSP system. *In vivo* imaging of a rat's abdominal blood flow is also presented. [DOI: 10.1117/1.1695409]

Keywords: optical Doppler tomography; digital signal processor.

Paper 44002 received Jun. 4, 2003; revised manuscript received Dec. 12, 2003; accepted for publication Dec. 19, 2003.

1 Introduction

Optical Doppler tomography (ODT)[1–3] is a noninvasive, noncontact imaging modality that combines coherent gating and a Doppler principle to obtain high-resolution cross-sectional images of tissue microstructure and blood flow simultaneously. As a functional extension of optical coherence tomography (OCT),[4] ODT performs depth-resolved tomographic flow measurements in a series of time frames. Since ODT imaging is highly localized, the image acquisition speed must exceed the time scale of the movement of nonstationary objects in order to avoid motion artifacts. In addition, a real-time ODT data-processing scheme that provides rapid and continuous imaging and monitoring of blood flow synchronized with structural imaging is needed for timely diagnosis. The development of a rapid-scanning optical delay line is the first technical challenge for real-time ODT, and it is practically solved.[5–7]

The second issue is the development of a real-time data-processing and display unit. In general, two steps of data processing are involved in simultaneous detection of structure and flow velocity using ODT. First, a structural cross-sectional (B-scan) image is obtained by segmenting the raw interference data to obtain consequent A-line profiles, and then mapping the logarithm of the amplitude of A-line profiles to gray scale. Second, a flow image is obtained using Doppler analysis and displayed either aside or overlapped on top of the structural image for simultaneous visualization of flow regions and directions. Formation of the structural image is in general straightforward and fast by using a Hilbert transform and envelope detection. Flow imaging, however, requires intensive computations.

Because of ODT's promising potential in clinical applications, real-time ODT is under extensive investigation. Westphal et al.[8] and Rollins et al.[9] employed autocorrelation schemes in their hardware to measure the phase of the correlation function, which corresponds to the local Doppler shift. Zhao et al.[10] and Ding et al.[11] proposed an optical Hilbert transformation method utilizing the polarization properties of light to obtain the phase information for measuring flow velocity. Yang et al.[12] implemented an in-phase and quadrature (I&Q) demodulator in their hardware and a subsequent Kasai autocorrelation velocity estimator in their software. All of these techniques focused on the hardware implementation of a particular Doppler algorithm in order to eliminate the need for offline processing of the recorded interference signals. Therefore, the Doppler algorithm implemented determines the performance of the velocity estimator.

The Doppler analysis algorithms currently used in ODT can be classified into three major categories: a short-time Fourier transform (STFT) method,[1–3] a filtering method,[13] and a Hilbert transform method.[14] The STFT method extracts local Doppler shifts by measuring the spectrum centroid displacement of the interference signal from the carrier modulation. The filtering method directly maps the frequency shift at each pixel using a digital filtering technique performed by a bank of narrow bandpass filters. The Hilbert transform method, also called the correlation method, calculates the autocorrelation of individual A-lines or the cross-correlation between sequential A-lines, and estimates velocity changes from the phase of the correlation function. The STFT method is efficient in computation for its simplicity, but it is generally noisy owing to stochastic modifications of the Doppler spectrum by fluctuating scatterer distributions in the flow field.[15] Compared with STFT and Hilbert transform techniques, the filtering method is more accurate and robust to noise; however, its digital filtering process requires intensive computation.[13] The Hilbert transform method is known to have the highest velocity sensitivity when the phase is calculated from the cross-correlation between sequential A-lines.[13,14] Nevertheless, it tends to be noisy in lower flow speed and/or lower signal-to-

Address all correspondence to Quing Zu, University of Connecticut, 371 Fairfield Road U11257, Storrs, Connecticut 06269. Tel: 860-486-5523; FAX: 860-486-2447; zhu@engr.uconn.edu

1083-3668/2004/$15.00 © 2004 SPIE

noise ratio (SNR) regions because the correlation techniques are more sensitive to local decorrelation of the flow field resulting from low SNR or the high-velocity gradient that is present in a low flow velocity region.[13,16] The Hilbert transform method is also computation intensive, owing to the correlation process involved. An investigation of the offline computation load of these three categories of algorithms revealed that approximately 15, 60, and 106 s were required to process the flow image with a typical dataset of 512×2048 by the three algorithm categories, respectively, on a Pentium III 800-MHz personal computer (PC).[13]

In the real-time ODT techniques mentioned earlier, the Hilbert transform technique was employed. A Hilbert transform accomplished in hardware has the advantage of simple implementation; however, it lacks flexibility when a different Doppler method is to be investigated. An ODT system based on a digital signal processor (DSP) module, on the other hand, has the advantage of embedding probably any desired high-performance Doppler algorithm. In addition, it has the flexibility to incorporate other functional imaging algorithms such as those needed for spectroscopic OCT.[17] In this paper, we present a real-time ODT system using a low-cost DSP module embedded with two advanced Doppler algorithms. In this approach, a custom-designed DSP is incorporated into a conventional ODT system with a grating-based rapid-scanning delay line. Configured with a signal-processing evaluation module and an analog-to-digital (A/D) daughterboard, this DSP performs data acquisition, structural imaging, and flow processing sequentially. A real-time data-processing capability is expected based on an evaluation of DSP performance. Experiments on Intralipid flow demonstrate that in M-scan mode this DSP system is able to faithfully capture pulsatile flow at a rate of several hundred pulses per minute. *In vivo* imaging of a rat's abdominal blood flow is also demonstrated. Two advanced velocity estimation algorithms, namely, weighted-centroid and sliding-window filtering, are embedded in this DSP, and the superior performance of the sliding-window filtering algorithm over the weighted-centroid technique as reported in our previous study is verified in real-time ODT imaging circumstances.

2 Flow Velocity Estimation Algorithms Embedded in the DSP

In our previous work,[13] we discussed five velocity estimation algorithms, among which we introduced two new techniques: weighted-centroid and sliding-window filtering. In that work, we assessed the velocity estimation accuracies of these algorithms by simulations. The algorithms were evaluated for different noise levels at the same velocity resolution. At a higher SNR, e.g., greater than 15 dB, the estimation errors of sliding-window filtering and weighted-centroid algorithms were 2 and 3%, respectively, while the previously existing centroid and correlation techniques gave errors of more than 4%. At a lower SNR of 6 dB, the estimation errors of sliding-window filtering and weighted-centroid algorithms were 11 and 15%, respectively, while the other centroid and correlation techniques had errors greater than 19%. The overall results for SNR from 5 to 20 dB demonstrated that the weighted centroid technique slightly outperformed previously available algorithms, and the sliding-window filtering technique was superior to all other techniques in accuracy of velocity estimation, most significantly at low signal-to-noise ratios. In terms of operational count, however, the computation time of the sliding-window filtering algorithm was six times longer than that of the weighted-centroid algorithm. To take advantage of the robust sliding-window filtering technique on flow estimation and also to satisfy the real-time data processing demand, we have designed a low-cost DSP-based signal-processing module. This DSP module improves the calculation speed dramatically so that real-time data processing is readily achieved. In Secs. 2.1 and 2.2 we give a brief review of these two algorithms embedded in the DSP.

Since ODT requires a broader bandwidth to detect frequency shifts that are due to moving scatterers, averaging of repeated data acquisition to improve SNR is necessary. By taking into account the averaging, we express the detected 2-D interference signal after Hilbert transform as[13]

$$\tilde{z}_{k,i}(t)=|\tilde{z}_{k,i}(t)|e^{j\omega t} \tag{1}$$

where t is the time argument along the depth scanning direction, k represents the lateral dimension of the k'th A-line, i is the index of the repeated A-line measurements, and ω is the angular frequency of signal reflected from the target. The sampled interferometric signal is represented by $\tilde{z}_{k,i}(t_n)$, where $t_n=nt_s$ and t_s is the sampling interval, and the power spectrum of $\tilde{z}_{k,i}(t_n)$ is denoted with $P^n_{k,i}(\omega)$.

2.1 *Weighted-Centroid Algorithm*

The weighted-centroid algorithm, which is an extension of the conventional centroid method, is given by

$$\overline{\omega}_{k,i}(n)=\int_{-\infty}^{\infty}\frac{\omega[P^n_{k,i}(\omega)]^{\xi}d\omega}{[P^n_{k,i}(\omega)]^{\xi}d\omega}, \tag{2}$$

where a positive integer ξ is introduced as the weight of the spectrum to emphasize the frequency component corresponding to the peak power in the spectrum.

2.2 *Sliding-Window Filtering Algorithm*

The sliding-window filtering technique directly maps the frequency shift at each pixel using a digital bandpass filtering. In this approach, the detected signal $\tilde{z}_{k,i}(t_n)$ at each depth scanning range t_n is processed by a filter bank consisting of M filters, $f_1, f_2,\ldots,f_M$, yielding one signal at the output of each filter as illustrated in Fig. 1. This produces M output signals $\tilde{O}_m(t_n)$, $m=\{1,2,...,M\}$ for each detected signal so that

$$\tilde{O}_m(t_n)=\tilde{z}_{k,i}(t_n)\otimes F^{-1}(f_m), \tag{3}$$

where the passband of f_m is $[(2m-1)\pi/2M-\Delta\omega/2,(2m-1)\pi/2M+\Delta\omega/2]$ with $\Delta\omega$ being the filter bandwidth. The output signals represent components of the input signal that change at a rate corresponding to the passband of the filter. Then, at each range t_n, the energy in each filter, $\varepsilon_m(t_n)$, is estimated as[16]

$$\varepsilon_m(t_n)=\sum_{t'=t_n-\Delta t/2}^{t_n+\Delta t/2}[\tilde{O}_m(t')]^2, \tag{4}$$

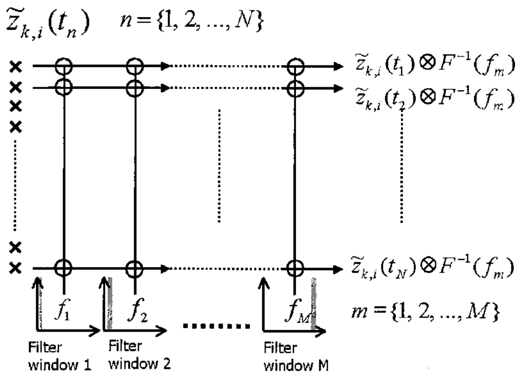

$\tilde{z}_{k,i}(t_n)$ $n=\{1,2,...,N\}$
$\tilde{z}_{k,i}(t_1)\otimes F^{-1}(f_m)$
$\tilde{z}_{k,i}(t_2)\otimes F^{-1}(f_m)$
$\tilde{z}_{k,i}(t_N)\otimes F^{-1}(f_m)$
f_1
f_2
f_M
$m=\{1,2,...,M\}$
Filter window 1
Filter window 2
Filter window M

Fig. 1 Schematic of the sliding-window filtering technique. The sliding-window filtering bank is used to measure the components of each interference signal that change at a rate corresponding to the passband of the filters.

where Δt represents a short-range window centered at t_n. At any given t_n, the filter window position corresponding to the maximum $\varepsilon_m(t_n)$, $m=\{1,2,...,M\}$, represents the most significant frequency component, equivalently the frequency shift, of flow signal. In this algorithm, a second-order Chebyshev filter of the first kind is chosen.[13]

3 Architecture of a DSP-Based ODT System

The DSP-based ODT system is constructed on top of a conventional ODT device.[19] Functionally, this new system consists of four units (shown in Fig. 2): the ODT scanner, a control signal generator, an embedded real-time processor, and the image display module. A description of the detailed function of each unit follows.

3.1 *Unit 1. ODT Scanner*

The ODT scanner is a typical balanced setup configured with one 1×2 and one 2×2 fiber coupler (see Fig. 3). The low-coherence source is a superluminescent diode with a 1300-nm center wavelength, 40-nm spectral width, and 2.0-mW maximum output power. In the reference arm, a scanning optical delay line based on Littrow mounting of a diffraction grating is used to generate range scanning and a phase modulation for carrier frequency as well.[13] In the sample arm, a galvanometer and an achromatic lens are integrated to perform the repeatable lateral scanning that is essential for continuous monitoring of flow. In the detection arm, the signal is detected differentially by an autobalanced receiver, with which the source intensity noise is rejected. The signal is then amplified, bandpass filtered, and digitized.

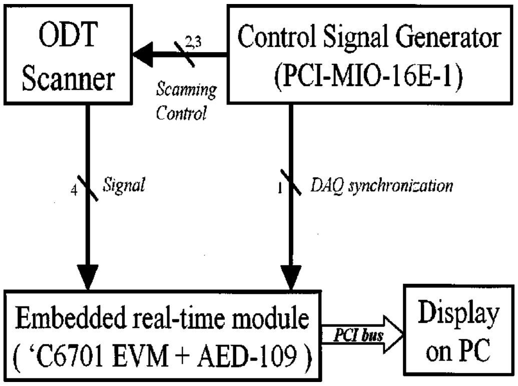

ODT Scanner
Control Signal Generator (PCI-MIO-16E-1)
2,3
Scanning Control
4
Signal
1
DAQ synchronization
Embedded real-time module ('C6701 EVM + AED-109)
PCI bus
Display on PC

Fig. 2 Schematic of the DSP-based ODT system.

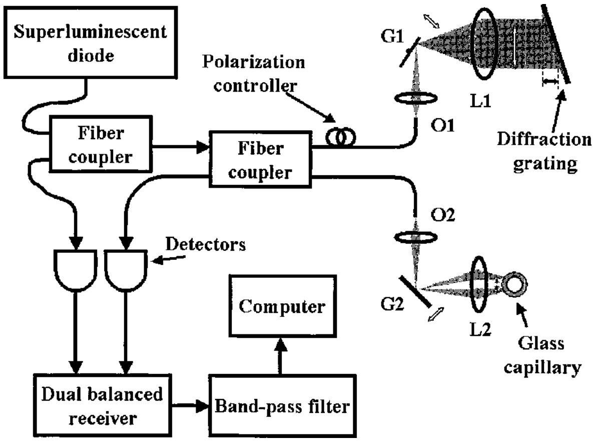


Fig. 3 The ODT scanner, which is a typical balanced setup with both axial and lateral scanning, performed by a galvanometer. O1, O2, microscope objectives; G1, G2, galvanometers; L1, L2, achromatic lenses.

Using this system, cross-sectional imaging of one frame per second is obtained by driving the reference arm and sample arm galvanometers with a 128-Hz triangle wave and a 1-Hz sawtoothed wave, respectively. The reference galvanometer is capable of scanning at a much higher frequency; however, a 128-Hz scan speed is used because of the bandwidth limitation of the dual-balanced receiver. Each frame consists of 128 axial scans containing 2048 data points at each round of scan. Because a 50% duty cycle of galvanometer scanning is used, each A-line covers 1024 points, and four consecutive A-lines are then averaged to improve the SNR. In the axial dimension, 16 points are averaged at one pixel, resulting in a pixel dimension of 32×64 covering an area 2.5×1.0 mm (Intralipid study) and 2.5×2.0 mm (*in vivo* study).

3.2 *Unit 2. Control Signal Generator*

To synchronize axial and lateral scanning as well as data acquisition, we built a control signal generator using LabVIEW based on a National Instruments (NI) multifunction data acquisition (DAQ) card, PCI-MIO-16E-1. As shown in Fig. 4, this unit generates synchronization signals for ODT scan and data acquisition: one fast wave for reference arm scanning and one slow-wave for lateral scanning as previously mentioned, and a TTL-compatible triggering signal for data acquisition and real-time processing.

3.3 *Unit 3. Embedded Real-Time Processor*

The newly embedded real-time processing unit is a combination of the TMS320C6701 DSP evaluation module ('C6701 EVM, Texas Instruments Inc.)[20–22] and an analog expansion daughterboard, AED-109[23] (Signalware Inc.). The 'C6701 EVM board is a complete DSP system based on a 'C6701 DSP chip operating up to 1336 million instructions per second (mips) with a central processing unit clock rate of 167 MHz. With 'C6701, one can actually perform two multiplication and

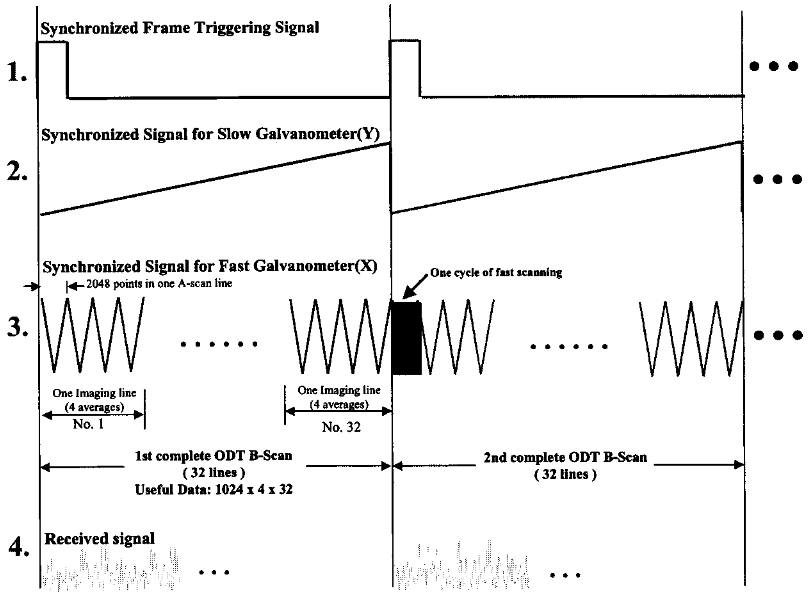


Fig. 4 The synchronization strategy of ODT scanning and data acquisition.

accumulation operations within a single clock cycle. This board also provides 256 kbytes of external 133-MHz synchronous-burst static RAM (SBSRAM) and 8 Mbytes of external 100-MHz synchronous dynamic RAM (SDRAM), both of which could be used as data buffers. The daughterboard AED-109 has two dual-channel 12-bit A/D converters with up to 8 MHz of sampling speed and two digital to analog (D/A) converters; the input channels have been configured into differential inputs for higher SNR consideration. In addition to analog channels, 16-buffered digital signals can be used as flags for system control.

The data flow with this real-time processor is shown in Fig. 5. When ODT scanning starts, the daughterboard AED-109 acquires the signal from the ODT scanner and the 12-bit data are buffered into a 256-word two-way first-in first-out (FIFO) unit in the field-programmable gate array (FPGA) XCV100E6 (Xilinx Inc.). As soon as this FIFO unit is full, the XCV100E6 triggers the 'C6701 DSP to transfer data to the 'C6701 in direct memory access (DMA) mode. The circular buffer in the 'C6701 DSP memory, which is segmented into eight frames with 256 words per frame, can hold 2048 words of data acquired during one complete fast scanning. The transferred 256-word data are saved in one frame of the circular buffer, then the buffer is polled to process the new incoming data. For structural image formation and flow calculation using the weighted-centroid algorithm, this DSP module processes each section (256 words) in the circular buffer continuously; however, for flow calculation using the sliding-window filtering algorithm, the DSP needs to acquire four continuous frames before it starts to process 1024 data points for a complete A-scan line. The processed structural image and flow image data are saved in each buffer in the SDRAM on the 'C6701 EVM. The next step is to load the image data into a personal computer for display.

3.4 *Unit 4. Image Display Module*

A real-time image display is the last yet critical step of this real-time DSP-ODT system. The real-time data exchange

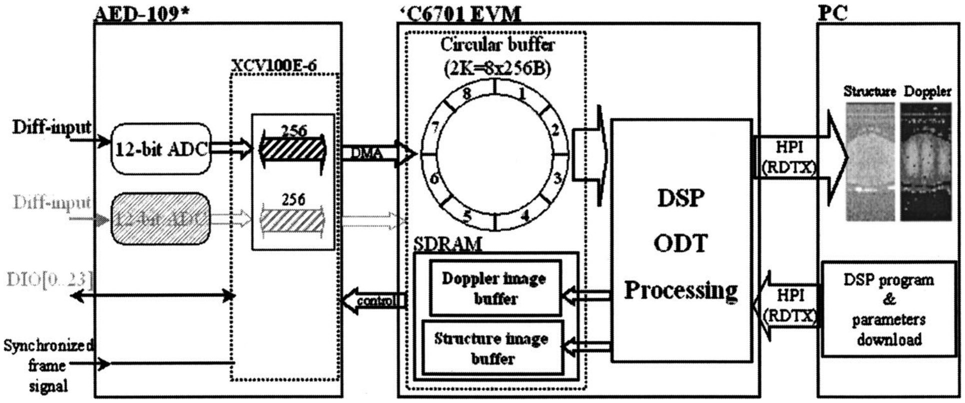


Fig. 5 Data flow in the DSP-ODT system: hardware perspective. An eight-frame circular buffer in the 'C6701 DSP continuously receives 12-bit data from the AED-109 analog daughterboard; the DSP module simultaneously processes data in each frame; a PC loads the image data right after processing through the host-port interface (HPI).

(RTDX) provided by Texas Instruments offers continuous bidirectional data exchange in real time with minimal perturbation to the application. However, in the 'C6701 EVM, the maximum RTDX rate is 20 kbytes/s. Since structural image and flow image data are in floating data format (4 bytes), a total of 96 kbytes of data need to be loaded in one image frame (1 s) for simultaneous display of both structural and flow images, which is beyond the RTDX speed of our current DSP. To overcome this problem, the host-port interface (HPI) between the DSP and PC is employed to load image data right after processing. With this technique, flow image processing and image uploading are performed alternately. During the operation, the DSP is continuously processing and saving image data in 8 Mbytes of SDRAM on the 'C6701 EVM to avoid data loss. After the DSP has completed the desired frames, all of the image data are loaded at one time. This strategy ensures an image acquisition speed up to a maximum of 128 A-scans per frame, which is the actual scan speed. If a photoreceiver with a broader bandwidth is employed in the ODT scanner to accommodate faster scanning, and a new high-speed RTDX (HS-RTDX) with a 2-Mbytes/s transfer rate is implemented, we expect to achieve much higher processing and display updating speed than the current 1-Hz frame rate.

4 Evaluation of DSP Performance

To study the performance of the DSP, we estimated the its cycles to three processing modes: structural image, weighted-centroid flow image, and sliding-window filtering flow image. The resulting operational structures for implementing each of them are shown in Fig. 6 and described in the following paragraphs.

For a structural image, one set of A-scan lines with 1024 data points is segmented uniformly into 64 sections, with 16 points per section. The data in each section are then Hilbert transformed by a floating-point fast Fourier transform (FFT) and IFFT pair.[24] For each section, the maximum of the absolute values of Hilbert-transformed data is determined, which represents the signal level in the local pixel. Therefore, a total of 64 pixel values are obtained for single A-scan data in structural imaging. Most calculations are FFT and IFFT pairs, resulting in 0.043 million cycles for the fixed-point calculation and 0.041 million cycles for a floating-point calculation.

For the weighted-centroid algorithm, again, one set of A-scan lines is segmented into 64 sections. The data in each section are zero-padded to 64 points, and then Fourier transformed by 64-point complex floating-point FFT. From the calculated 32-point spectrum value of each section, the Doppler frequency shift is calculated by finding the local center frequency with the weighted-centroid principle and subtracting the known modulation frequency. Again, a total of 64 Doppler shift values are obtained for single A-scan data in flow imaging by the weighted-centroid technique. Most calculations focus on FFT. By using complex radix-2 floating-point FFT, about 0.096 million cycles are needed to finish the calculation of one 1024-point A-scan datum. If complex radix-4 fixed-point FFT is implemented instead, only 0.077 million cycles are needed.

For a sliding-window filtering algorithm, a complete set of A-scan lines with 1024 data points is filtered individually by 32 second-order IIR bandpass filters, each of which has one of 32 evenly distributed narrow passbands from 0 to π, and the passbands of all filters cover the entire spectrum. The number of 32 evenly distributed filters is chosen to maintain the same velocity resolution as the weighted-centroid does where the

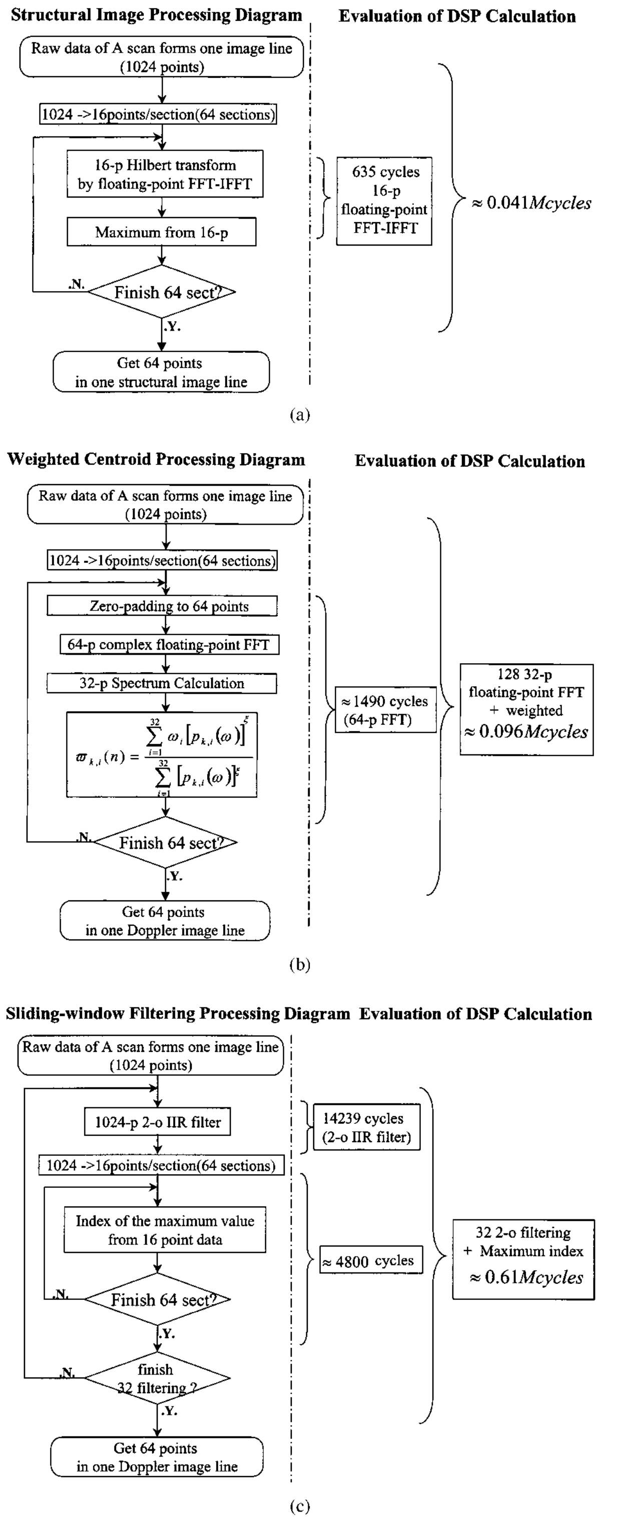

Structural Image Processing Diagram
Evaluation of DSP Calculation
Raw data of A scan forms one image line (1024 points)
1024 ->16points/section(64 sections)
16-p Hilbert transform by floating-point FFT-IFFT
Maximum from 16-p
.N.
Finish 64 sect?
.Y.
Get 64 points in one structural image line
635 cycles 16-p floating-point FFT-IFFT
≈ 0.041Mcycles
(a)
Weighted Centroid Processing Diagram
Evaluation of DSP Calculation
Raw data of A scan forms one image line (1024 points)
1024 ->16points/section(64 sections)
Zero-padding to 64 points
64-p complex floating-point FFT
32-p Spectrum Calculation
$\varpi_{k,i}(n) = \frac{\sum_{i=1}^{32} \omega_i [p_{k,i}(\omega)]^{\xi}}{\sum_{i=1}^{32} [p_{k,i}(\omega)]^{\xi}}$
.N.
Finish 64 sect?
.Y.
Get 64 points in one Doppler image line
≈ 1490 cycles (64-p FFT)
128 32-p floating-point FFT + weighted ≈ 0.096Mcycles
(b)
Sliding-window Filtering Processing Diagram
Evaluation of DSP Calculation
Raw data of A scan forms one image line (1024 points)
1024-p 2-o IIR filter
1024 ->16points/section(64 sections)
Index of the maximum value from 16 point data
.N.
Finish 64 sect?
.Y.
.N.
finish 32 filtering ?
.Y.
Get 64 points in one Doppler image line
14239 cycles (2-o IIR filter)
≈ 4800 cycles
32 2-o filtering + Maximum index ≈ 0.61Mcycles
(c)

Fig. 6 Code structures of three processing modes: (a) Structural imaging, (b) weighted-centroid flow imaging, (c) sliding-window filtering flow imaging.

Table 1 Calculation cycle evaluation of three processing modes based on the 167-MHz ‘C6701 EVM.

Processing Mode	Fixed-point (million cycles)	Floating-point (million cycles)
Structural	0.043 (0.26 ms)	0.041 (0.25 ms)
Weighted-centroid	0.077 (0.47 ms)	0.096 (0.58 ms)
Sliding-window filtering	0.32 (1.9 ms)	0.61 (3.7 ms)

Table 2 DSP calculation accuracy of fixed-point and floating-point calculations compared with a Matlab calculation in 8-byte double data format. The number is the mean square error with respect to Matlab results.

Processing Mode	Fixed-point (2-byte short)	Floating-point (4-byte float)
Structural	3.5×10^{-2}	1.5×10^{-6}
Weighted-centroid	6.8×10^{-3}	3.4×10^{-7}
Sliding-window filtering	3.9×10^{-2}	6.1×10^{-4}

spectrum profile consists of 32 points. The filtered 1024 data are segmented into 64 sections equally, with 16 points per section. In each section, the filter number corresponding to the maximum signal power after these 32 filters is indexed as the center frequency, and the Doppler shift is calculated by subtracting the modulation frequency. The resulting total number of Doppler shift values for single A-scan data in the sliding-window filtering method is 64. Most calculations are IIR filtering banks. The execution takes about 0.32 million cycles for 1024-point fixed-point calculations, and 0.61 million cycles for floating-point calculations. As shown in Table 1, the most intensive calculation is for the sliding-window filtering technique, which is about six times that of the weighted-centroid technique. Nevertheless, for one A-line at a duration of 1/128 s, the computation load of all three modes is within the 1.3-million cycle real-time limitation of the 167-MHz ’C6701 EVM.

The above-described DSP speed estimate is based on the assumption that the DSP has enough internal memory. However, the ’C6701 DSP provides only 64 kbytes of internal memory for program storage and 64 kbytes of internal memory for data storage. As a result, the relatively low-speed external memory in SBSRAM and SDRAM has to be used for the DSP object file because it has a size of 95 kbytes, which exceeds the 64-kbyte internal program memory. If a code is present in an external memory, more cycles are needed for its execution. In addition, a large number of data buffers are needed in external memory. By optimizing memory allocation and code efficiency, we have achieved continuous calculation for simultaneous structural and flow imaging in less than 1 s with the current DSP module.

To verify that our DSP code is efficient, we evaluated the ratio of DSP speed for two flow-processing modes and compared it with Matlab calculations. In Matlab, 2.5 and 17 s are needed for the weighted-centroid technique and the sliding-window filtering technique, respectively, on a Pentium-4 2.0-GHz personal computer. The ratio of speed in Matlab is 6.8, which is very close to the 6.4 achieved by the ’C6701 DSP using the floating-point calculation.

The ’C6701 is a floating-point DSP, and it can operate in either fixed-point or floating-point calculation mode. The ’C6701 uses a 2-byte short data type in fixed-point FFT and IIR functions and a 4-byte floating data type in floating-point FFT and IIR functions. Because Matlab uses an 8-byte double data type to do the calculation, we can use the result from Matlab as a benchmark, and the calculation accuracy of fixed-point and floating-point calculations can be found by studying the mean-square-error (MSE) with respect to the Matlab results. Table 2 lists the MSE in the DSP calculation for fixed-point and floating-point data types. By comparing Tables 1 and 2, we can see that there is a tradeoff between speed and calculation accuracy in choosing either a fixed-point or a floating-point DSP. Since our DSP system is able to process flow information in the desired time frame, the floating-point DSP is favored for its higher accuracy and is therefore implemented in our code.

5 Results

5.1 *Experimental Flow*

To demonstrate the performance of our DSP-ODT system, we constructed an experimental flow setup with 0.5% Intralipid solution. The ODT probing beam was focused to the axis of a glass capillary tube with a 1.0-mm inner diameter, 1.2-mm outside diameter, and 100-mm length. The glass capillary was mounted to a rotary stage and connected to a 2-m-long 3/32-in. plastic tubing. The mean flow velocity of Intralipid through the glass capillary tubing was converted from the time interval of liquid traveling 1 m in the 3/32-in. plastic tubing, which was placed straight on a horizontal plane. A peristaltic tubing pump was used to drive the Intralipid through the capillary tubing and to circulate it in the tubing loop. The pulsation induced by peristaltic pumping was used to emulate *in vivo* blood circulation. The tubing pump gave approximately 3 pulses per second (pps) at its minimum rotation speed.

Using this setup, 30 frames were taken continuously at a 1-Hz frame rate to perform M-mode scans. During this 30-frame interval, the pumping speed was manually changed, with a sequence of no flow, minimum speed, gradually increased speed, gradually decreased speed, increasing/decreasing again, and stopping. One frame of the real-time acquired and processed flow images is shown in Fig. 7. In those flow images, several color strips are distributed perpendicular to the lateral scanning direction in the Intralipid flow region. The axial and transverse flow profiles are also plotted, where the axial profile gives a clear laminar flow pattern within a single color strip corresponding to one pulse of flow, and the transverse profiles reveal the pulsation of the flow. The number of flow cycles and the magnitude of frequency shift correspond very well with the numbers of pulses generated by the pump and the speed of the flow. With this DSP-

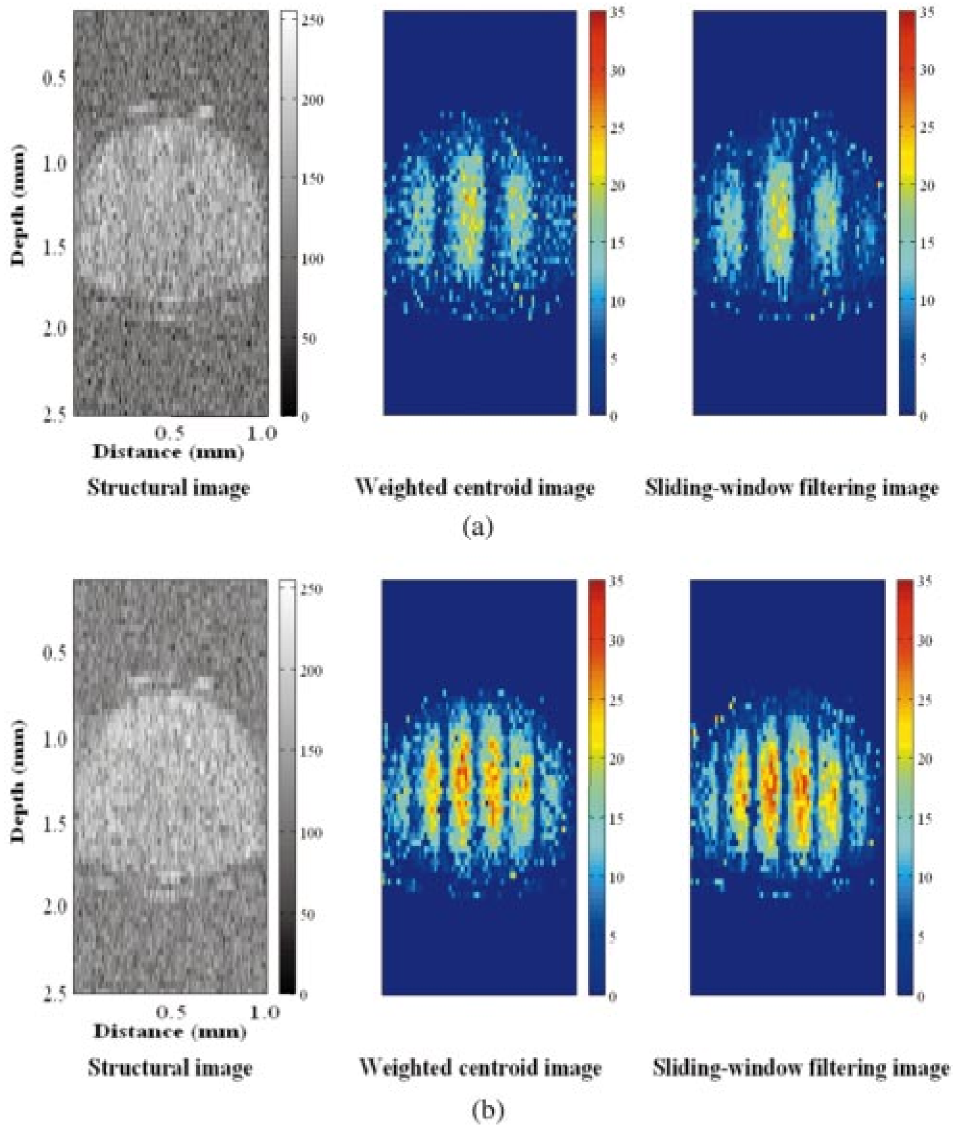


Fig. 7 Experimental results on pulsatile Intralipid flow. (a) 3 pulses per second, (b) 6 pulses per second.

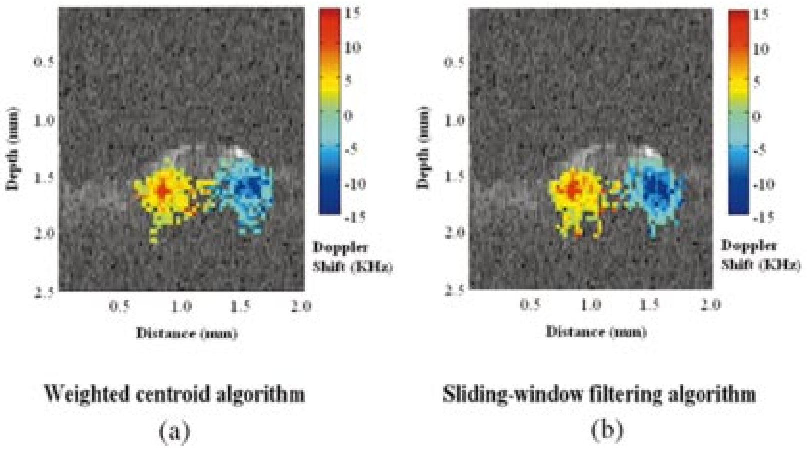
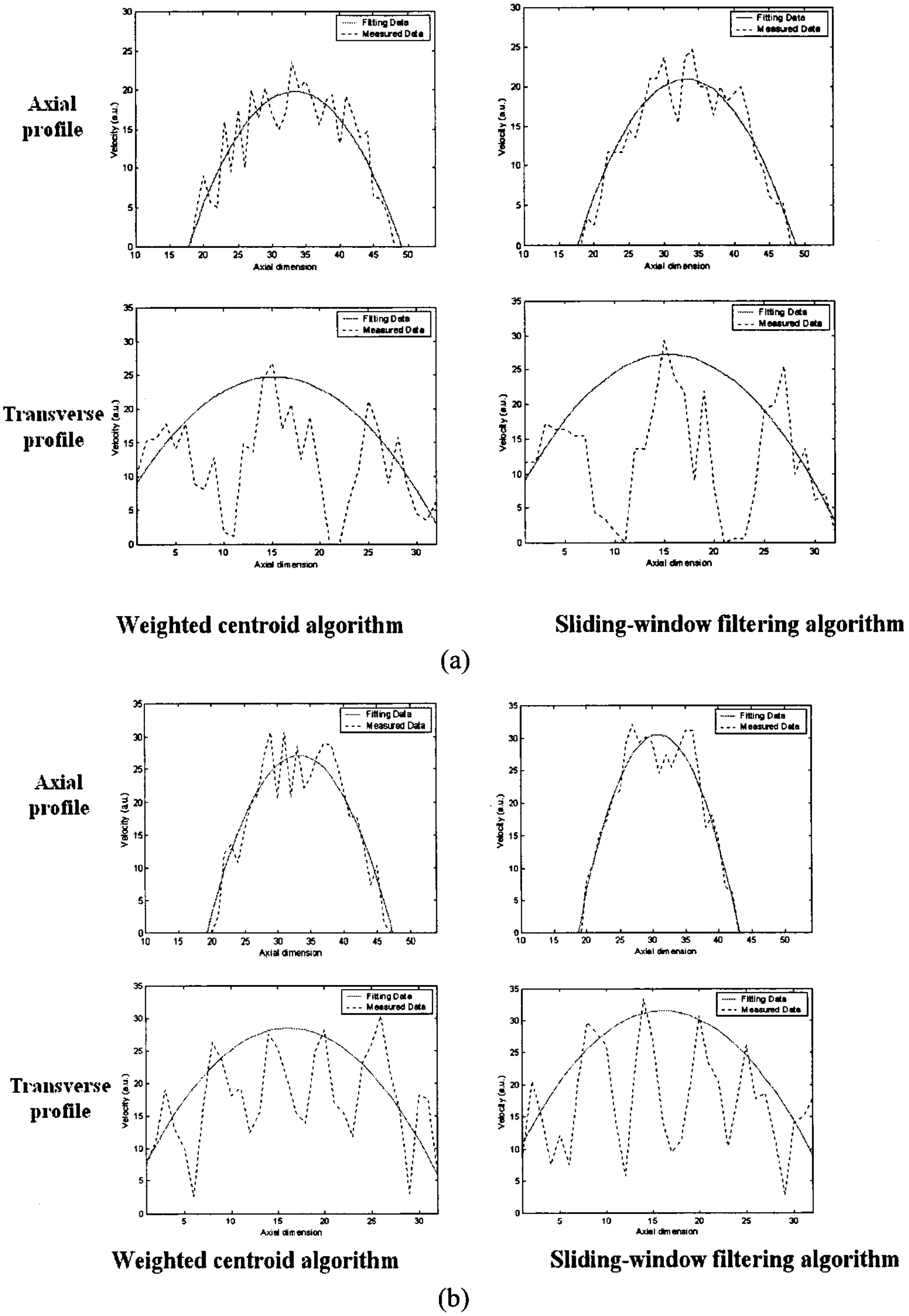


Fig. 9 *In vivo* blood flow imaging of the abdominal blood vessels of a small rat. Two bidirectional blood vessels are clearly identified. (a) Weighted-centroid flow image superimposed on a structural image. (b) Sliding-window filtering flow image superimposed on a structural image.



Weighted centroid algorithm

Sliding-window filtering algorithm

(a)

Weighted centroid algorithm

Sliding-window filtering algorithm

(b)

Fig. 8 Axial and transversal velocity profiles by weighted-centroid algorithm and sliding-window filtering algorithm for pulsatile Intralipid flow. (a) 3 pulses per second, (b) 6 pulses per second.

ODT system, we were able to faithfully capture a dynamic Intralipid flow of as high as 10 pps. This suggested that our system was capable of monitoring pulsatile flow circulation in humans and experimental animals.

To investigate the performance of the two embedded Doppler algorithms in this real-time processing scenario, we compared the flow velocity estimated by the weighted-centroid technique and the sliding-window filtering technique at 3 and 6 pps (shown in Fig. 8). The mean flow speeds were 50 mm/s at 3 pps and 55 mm/s at 6 pps, and the Doppler angle was set at 80 deg. The maximum Doppler shifts produced in the laminar flow were 26.7 kHz and 29.4 kHz for 3 pps and 6 pps, respectively. It can be seen from Fig. 7 that the sliding-window filtering method gives a much less noisy, more accurate flow profile, and a clearer pulsation pattern than the weighted-centroid algorithm does. This result is consistent with our simulations and experiments reported in Ref. 13, which also indicates that flow velocity can be quantified with the prior knowledge of the Doppler angle in this real-time detection.

5.2 *In vivo study*

We have conducted *in vivo* study on an animal to validate the real-time imaging capability of our DSP-based ODT system. A 78-g rat was prepared and operated upon under the protocol approved by the University of Connecticut (Z2400201). We imaged two blood vessels running parallel to each other inside the opened abdomen, and euthanized the animal at the end of the study. One frame of the processed structural and flow images is shown in Fig. 9. In this figure, the two blood vessels are clearly identifiable as carrying bidirectional blood flow. However, the color strips representing the pulsatile flow seen in the Intralipid experiment were not observed in this *in vivo* imaging, although the image frame rate was several times higher than the rat's pulse rate, which was emulated in the Intralipid study. One factor that accounts for the missing color strips could be the damping of pulsatile flow because the blood vessels imaged were less than 1 mm. Nevertheless, this *in vivo* study demonstrated that our system was capable of monitoring *in vivo* bidirectional blood flow in realtime.

6 Discussion and Summary

For the two velocity estimation algorithms that were embedded in the DSP module, the velocity resolution was determined either by the length of the FFT for the weighted-centroid algorithm, or the filter bandwidth for the sliding-window filtering algorithm. Better velocity resolution leads to higher sensitivity. However, there is a tradeoff between sensitivity and the robustness to noise that affects the accuracy of the estimate, and another tradeoff between sensitivity and dynamic range, or the maximum detectable velocity. We have chosen the FFT length or the amount of filters to be 32, which was based on our previous study to find a compromise between sensitivity and the robustness to noise for the dynamic range of the blood flow of the subject.

In our DSP-based ODT system, we have the flexibility of modifying system parameters, which can easily be done through the front panel of LabVIEW. We also have the flexibility of incorporating different flow algorithms in the DSP processing module because the client program in the DSP has been modularized into the DAQ module, the processing module, and the communication module. It is easy to implement new algorithms in the processing module without changing the DSP code structure. Instead of implementing the DSP code in hardware, we have a host program coded in LabVIEW, which downloads a DSP object file to the 'C6701 EVM before the ODT system starts, and triggers the DSP after initializing the DSP-ODT system parameters. Therefore, any new modifications to the algorithms can easily be implemented in software.

The performance of this DSP system can be further improved if a DSP board with a higher system clock or multiple DSP boards can be employed. Further improvements in the performance of this DSP system will produce a higher scanning speed, which, however, will slightly affect the flow imaging performance. In our scanning optical delay line, the phase modulation frequency and the scanning speed can be separately controlled. However, if the scanning speed is greatly increased, the phase modulation frequency must be increased as well to maintain enough numbers of interference fringes to be detected. If the Nyquist rate of sampling is increased accordingly, the length of the FFT for the weighted-centroid algorithm or the amount of filter for the sliding-window filtering algorithm has to be adjusted to maintain the same sensitivity as in the current DSP setup. The adjustment of the FFT length or the filter amount will thus affect the accuracy of the estimate.

In summary, we have designed a DSP-based ODT module that features real-time data acquisition and processing by embedding two advanced Doppler analysis algorithms. This system incorporates a low-cost custom-designed DSP module in a conventional ODT system. Experiments using Intralipid flow and an *in vivo* study have demonstrated the real-time capability of this system to capture and monitor pulsatile flows.

Acknowledgments

The authors acknowledge partial funding support from the National Institutes of Health (NIH 1R01 DE11154-03). We also greatly appreciate the help provided by Dan Schwartz at Animal Services of the University of Connecticut in animal preparation.

References

1. X. J. Wang, T. E. Milner, and J. S. Nelson, "Characterization of fluid flow velocity by optical Doppler tomography," *Opt. Lett.* **20**, 1337–1339 (1995).
2. Z. P. Chen, Y. H. Zhao, S. M. Srinivas, J. S. Nelson, N. Prakash, and R. D. Frostig, "Optical Doppler tomography," *IEEE J. Sel. Top. Quantum Electron.* **5**, 1134–1142 (1999).
3. J. A. Izatt, M. D. Kulkarni, S. Yazdanfar, J. K. Barton, and A. J. Welch, "*In vivo* bidirectional color Doppler flow imaging of picoliter blood volumes using optical coherence tomography," *Opt. Lett.* **22**, 1439–1441 (1997).
4. D. Huang, E. A. Swanson, C. P. Lin, J. S. Schuman, W. G. Stinson, W. Chang, M. R. Hee, T. Flotte, K. Gregory, C. A. Puliafito, and J. G. Fujimoto, "Optical coherence tomography," *Science* **254**, 1178–1181 (1991).
5. G. J. Tearney, B. E. Bouma, and J. G. Fujimoto, "High-speed phase- and group-delay scanning with a grating-based phase control delay line," *Opt. Lett.* **22**, 1811–1813 (1997).
6. A. M. Rollins, M. D. Kulkarni, S. Yazdanfar, R. Ung-arunyawee, and J. A. Izatt, "*In vivo* video rate optical coherence tomography," *Opt. Express* **3**, 219–229 (1998).
7. N. G. Chen and Q. Zhu, "Rotary mirror array for high-speed optical coherence tomography," *Opt. Lett.* **27**, 607–609 (2002).
8. V. Westphal, S. Yazdanfar, A. M. Rollins, and J. A. Izatt, "Real-time, high velocity-resolution color Doppler optical coherence tomography," *Opt. Lett.* **27**, 34–36 (2002).
9. A. M. Rollins, S. Yazdanfar, J. K. Barton, and J. A. Izatt, "Real-time *in vivo* color Doppler optical coherence tomography," *J. Biomed. Opt.* **7**, 123–129 (2002).
10. Y. Zhao, Z. Chen, Z. Ding, H. Ren, and J. S. Nelson, "Real-time phase-resolved functional optical coherence tomography by use of optical Hilbert transformation," *Opt. Lett.* **27**, 98–100 (2002).
11. Z. Ding, Y. Zhao, H. Ren, J. S. Nelson, and Z. Chen, "Real-time phase-resolved optical coherence tomography and optical Doppler tomography," *Opt. Express* **10**, 236–245 (2002).
12. V. X. D. Yang, M. L. Gordon, A. Mok, Y. Zhao, Z. Chen, R. S. C. Cobbold, B. C. Wilson, and I. A. Vitkin, "Improved phase-resolved optical Doppler tomography using the Kasai velocity estimator and histogram segmentation," *Opt. Commun.* **208**, 209–214 (2002).
13. D. Piao, L. Otis, N. K. Dutta, and Q. Zhu, "Quantitative assessment

of flow velocity estimation algorithms for optical Doppler tomography imaging,'' *Appl. Opt.* **41**, 6118–6127 (2002).

14. Y. H. Zhao, Z. P. Chen, C. Saxer, S. H. Xiang, J. F. de Boer, and J. S. Nelson, ''Phase-resolved optical coherence tomography and optical Doppler tomography for imaging blood flow in human skin with fast scanning speed and high velocity sensitivity,'' *Opt. Lett.* **25**, 114–116 (2000).
15. M. D. Kulkarni, T. G. van Leeuwen, S. Yazdanfar, A. J. Welch, and J. A. Izatt, ''Velocity-estimation accuracy and frame-rate limitations in color Doppler optical coherence tomography,'' *Opt. Lett.* **23**, 1057–1059 (1998).
16. P. C. Li, C. J. Cheng, and C. K. Yeh, ''On velocity estimation using speckle decorrelation,'' *IEEE Trans. Ultrason. Ferroelectr. Freq. Control* **48**, 1084–1091 (2001).
17. Jian Ye, A. W. Schaefer, D. L. Marks, J. J. Reynolds, and S. A. Boppart, ''Real-time processing of spectroscopic optical coherence tomography,'' in *Proceedings of IEEE International Symposium on Biomedical Imaging*, TP-P1.22, Institute of Electrical and Electronic Engineers, New York (2002).
18. J. R. Crowe, B. M. Shapo, D. N. Stephens, D. Bleam, M. J. Eberle, E. I. Cèspedes, C. C. Wu, D. W. M. Muller, J. A. Kovach, R. J. Lederman, and M. O'Donnell, ''Blood speed imaging with an intraluminal array,'' *IEEE Trans. Ultrason. Ferroelectr. Freq. Control* **47**, 672–681 (2000).
19. D. Piao and Q. Zhu, ''Quantifying Doppler angle and mapping flow velocity by a combination of Doppler-shift and Doppler-bandwidth measurements in optical Doppler tomography,'' *Appl. Opt.* **42**, 5158–5166 (2003).
20. Rulph Chassaing, *DSP Application Using C and the TMS320C6x DSK*, Wiley, New York (2002).
21. Texas Instruments, Inc., TMS320C62x DSP library Programmer's Reference (April 2002).
22. Texas Instruments, Inc., TMS320C67x FastRTS library Programmer's Reference (Oct 2002).
23. Signalware Co., ''Documentation Package for AED-109 Multi-Channel Analog Expansion Daughterboard,'' Ver. 1.0 (Oct. 2001).
24. N. Kalouptsidis, *Signal Processing Systems: Theory and Design*, Wiley, New York (1997).