SIViP
DOI 10.1007/s11760-013-0511-9

ORIGINAL PAPER

VLSI architecture for 4-D depth filtering

Arjuna Madanayake · Randeel Wimalagunarathne · Donald G. Dansereau · Renato J. Cintra · Len T. Bruton

Received: 6 June 2012 / Revised: 8 March 2013 / Accepted: 6 June 2013

Abstract We propose the application of light field cameras and depth-selective 4-D IIR filtering to enable video surveillance, leveraging the post-capture depth-selective filtering enabled by computational photography. Novel ultralow-complexity differential-form depth-selective 4-D IIR filter algorithms and their corresponding architectures are proposed for processing 4-D light fields. Practical results are presented for real-world video sequences, and a CMOS VLSI implementation of the arithmetic processing elements is synthesized. The architecture shows 86.66, 78.94 % reduction in multipliers and adders compared to direct-form structure and delivers 26 frames/s for light fields of size $16\times16\times128\times128$.

Keywords 4-D filtering · Light fields · Video imaging and surveillance · Plenoptic · Digital architecture

This work was supported by The University of Akron, Ohio, USA; CNPq and FACEPE, Brazil; and NSERC, Canada.

A. Madanayake (✉) · R. Wimalagunarathne
Department of Electrical and Computer Engineering, University of Akron, Akron, OH, USA
e-mail: arjuna@uakron.edu

D. G. Dansereau
Australian Centre for Field Robotics, School of Aerospace, Mechanical and Mechatronic Engineering, University of Sydney, Sydney, NSW, Australia
e-mail: d.dansereau@acfr.usyd.edu.au

R. J. Cintra
Signal Processing Group, Departamento of Estatística, Universidade Federal de Pernambuco, Recife, PE, Brazil
e-mail: rjdsc@dsp.ufpe.org

L. T. Bruton
Department of Electrical and Computer Engineering, University of Calgary, Calgary, AB, Canada
e-mail: bruton@ucalgary.ca

1 Introduction

For many real-time video surveillance and security applications, it is useful to selectively enhance or attenuate specific video objects over a specific range of depths. Such methods could employ multiple cameras to fully recover otherwise partly occluded objects. As an example, all single-camera views of a target that is hidden behind foreground objects (such as foliage) may be heavily occluded by the cover objects. In this context, 2-D aperture arrays, known as plenoptic or light field cameras [1,8,12,13,17,22,22] can overcome the hindrances of such occlusions, being employed in computational photography for algorithmic refocusing [8,16] and also have applications in visual odometry, and video stabilization [7,19].

In previous surveillance-related work [10,16,21], depth-selective filtering, tracking, and target reconstruction have been proposed. Furthermore, the methods proposed by Vaish et al. [21] yield impressive results, but are nonlinear and less well suited to hardware implementation. Our proposed hardware implementation is based on recursive linear methods and is considerably more compact than could be achieved utilizing their technique. The novel contributions of this paper are: (i) a low-complexity digital hardware architecture for 4-D IIR light-field-based depth filters using discrete spatial differentiators; and (ii) a digital hardware realization of the proposed low-complexity differential-form signal flow graphs allowing real-time video depth filtering.

2 Application of light fields for surveillance in the presence of occlusion

First-order digital filters have recently been proposed as a building block in light field processing [4,6,15]. In this

Published online: 04 July 2013

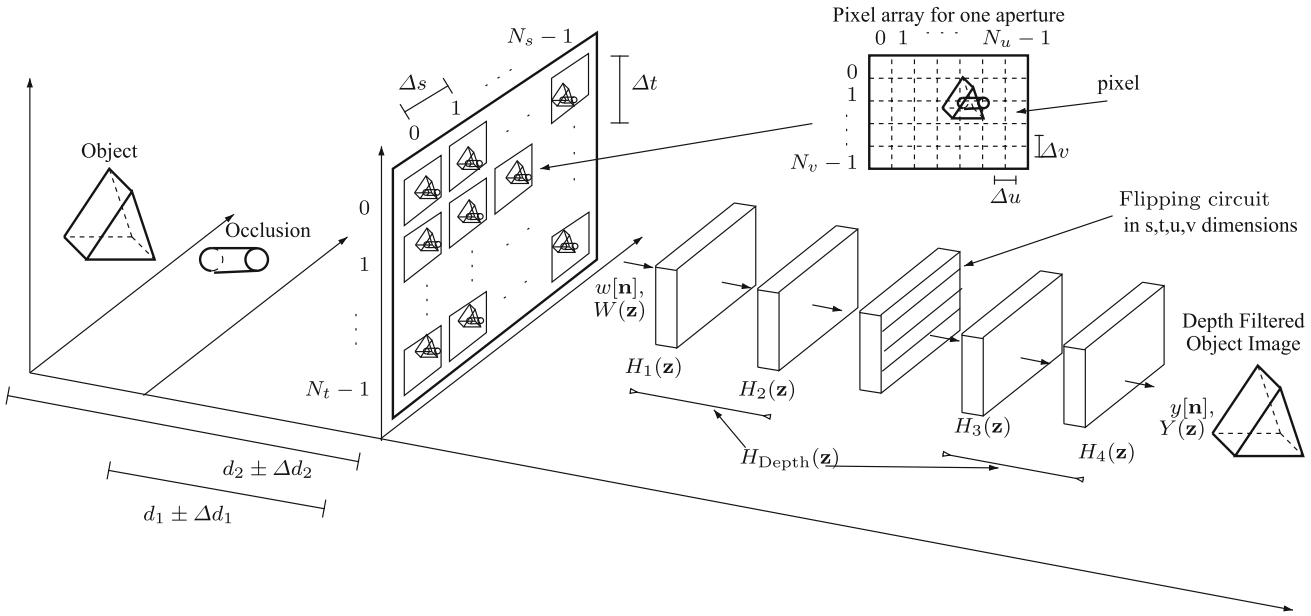

Fig. 1 From *left* to *right*: Object to be observed; occlusion; $N_s \times N_t$ array of apertures, each capturing $N_u \times N_v$ pixels; input 4-D light field $w[\mathbf{n}]$ or $W(\mathbf{z})$ in Z-domain; 4-D IIR frequency-hyperplanar filtering structure ($H_i(\mathbf{z})$, $i = 1, 2, 3, 4$); filtered 4-D output light field $y[\mathbf{n}]$ or $Y(\mathbf{z})$ in Z-domain, with attenuated stopband and enhanced passband objects; and an image rendered from the 4-D output light field. In the top right corner, an individual pixel array is detailed. Here $i = 1, 2$ correspond to the first depth enhancement stage, followed by a second depth enhancement stage $i = 3, 4$ which yields doubled selectivity as well as zero-phase filtering

paper, we focus on the efficient digital VLSI implementation of 4-D infinite impulse response (IIR) filters, with particular emphasis on the so-called depth filters, targeting applications in video/image processing for security and surveillance. A 3-D spatial scene is captured in real time by an array of video cameras as depicted in Fig. 1. The objective is to observe—without occlusions—the objects located at depth range $d_2 \pm \Delta d_2$ which constitute the "passband" depth range, despite the presence of objects located at undesired (i.e., stopband) depths, e.g., $d_1 \pm \Delta d_1$. Naturally, some objects located closer to the camera array occlude those located farther away. The proposed surveillance scheme removes undesired scene elements, including occluders, using 4-D linear filtering of the captured light fields. The proposed filters utilize a newly proposed low-complexity architecture based on 4-D extensions of differential-form signal flow graphs [3]. It follows from light field filter theory [6] that the output signal from the 4-D filter is also a light field, ideally containing only objects located within the depth range $d_2 \pm \Delta d_2$ with occluding objects attenuated. *This is equivalent to "looking behind" the objects located closer to the camera with unobstructed views of the spatial scene located at the desired depth despite heavy occlusions*.

The passband depth of the filter can be selected using a variety of methods. The simplest scenario is one in which the depth to be enhanced is known *a priori*. This is the case in many high-security applications such as surveillance of passengers in an airport. In scenarios in which the desired scene content has dynamic and unpredictable depth, an automated depth estimation system becomes necessary. This is a well-explored area, and several established methods for depth estimation and tracking from computer vision can be applied [9,11]. Yet another scenario allows for a human operator to adaptively tune the filter to allow manual, interactive, depth-selective surveillance. Such "person-in-the-loop" systems are especially important when critical decisions need to be made. We envision a hybrid system in which a human operator is interactively informed by machine vision depth estimates.

A video sequence of light fields is a 5-D signal consisting of time and four independent spatial dimensions: s, t, u, and v. Dimensions s, t are associated with aperture position, while dimensions u, v index pixels for a specific aperture. The scene is comprised of an arbitrary collection of pass band and stopband objects, and the camera is modeled as a zero-indexed rectangular array of $N_s \times N_t$ apertures, each with a zero-indexed array of $N_u \times N_v$ pixels. Sample spacings are given by Δs, Δt, Δu and Δv. *Mutatis mutandis*, the pixel sampling point locations are expressed by $n_u \Delta u$ and $n_v \Delta v$. Figure 1 illustrates these concepts.

The sensor pixels are captured and sampled by a 4-D real continuous domain light field input denoted by $w_a(s, t, u, v)$. The input 4-D light field is sampled and quantized furnishing the discrete 4-D light field:

$$w[\mathbf{n}] = Q\left(w_a(n_s\,\Delta s, n_t\,\Delta t, n_u\,\Delta u, n_v\,\Delta v,)\right), \tag{1}$$

where $\mathbf{n} = \begin{bmatrix} n_s & n_t & n_u & n_v \end{bmatrix}^\top$, $Q(\cdot)$ is the quantization operation which is usually rounding or truncation. Typically, the function $Q(\cdot)$ returns 8 bits per color field for a practical surveillance application with additive white Gaussian noise (AWGN) having variance $\sigma^2 = \frac{\Delta^2}{12}$ where $\Delta = 2^{-N}$ and N is the number of bits.

We aim at providing a digitally processed discrete light field $y[\mathbf{n}]$, obtained by submitting the input discrete light field $w[\mathbf{n}]$ to a series of purpose-designed 4-D IIR frequency-hyperplanar digital filters [5]. Filtering operations are performed on a per light field "4-D frame" basis, with no filtering occurring along the temporal dimension.

3 Proposed differential filter architecture

A complete description of the theory of passive multi-dimensional network resonance can be found in [4], and 4-D hyperplanar filters are summarized in [6,15]. The method for elemental pre-distortion of multi-dimensional networks leading to differential-form signal flow graphs was proposed in [3]. These filters are structurally and practical-BIBO stable because they are designed using the bilinear transform of pseudo-passive first-order self-resonant multi-dimensional prototype s-domain transfer functions [2,4]. The 3 dB bandwidth measured from the resonant plane in multi-dimensional frequency space is given in [4] as

$$B = \frac{R_0}{\|\mathbf{L}\|}, \tag{2}$$

where $\|\cdot\|$ returns the Euclidean norm of its argument and $\mathbf{L} = \begin{bmatrix} L_s & L_t & L_u & L_v \end{bmatrix}^\top$ is an inductance vector. Standard algebraic manipulations lead to the proposed *differential-form* version of the first-order 4-D frequency-hyperplanar prototype complex transfer function [6,15], obtained by applying the "elemental pre-distortion method" in [3] to the 4-D case, leading to

$$T(\mathbf{s}) = K_0 \cdot \frac{R}{R + \sum_{i\in\{s,t,u,v\}}(L_i s_i - r_i)}, \tag{3}$$

where $K_0 = R_0/R$, $R = R_0 + r_s + r_t + r_u + r_v$, and $-r_i$ are negative resistances that cancel the positive resistances r_i, leaving the denominator unchanged. By extending the method of elemental pre-distortion [3] into 4-D, selecting $r_i = L_i$, and applying the 4-D bilinear transform [4] to (3), we obtain the required form

$$H_{\mathrm{k}}(\mathbf{z}) = \frac{1}{1 + \sum_{i\in\{s,t,u,v\}} \alpha_i \frac{z_i^{-1}}{1+z_i^{-1}}} \equiv \frac{Y_k(\mathbf{z})}{W_k(\mathbf{z})}, \tag{4}$$

where $\alpha_i = -2L_i/R$, $k \in \{1,2,3,4\}$, and $W_k(\mathbf{z})$ and $Y_k(\mathbf{z})$ are the $\mathbf{z}$-transform of the input and output per filter stage, respectively. Also $W_1(\mathbf{z}) = W(\mathbf{z})$ and $Y_4(\mathbf{z}) = Y(\mathbf{z})$. The closed-form 4-D frequency response can be found by evaluating (4) over the 4-D unit hyper-circle $\mathbf{e} \equiv \begin{bmatrix} e^{-j\omega_s} & e^{-j\omega_t} & e^{-j\omega_u} & e^{-j\omega_v} \end{bmatrix}^\top$. This leads to the magnitude response function given by

$$|H_{\mathrm{k}}(\mathbf{e})| = \left| \frac{1}{1 + \sum_{i\in\{s,t,u,v\}} \alpha_i \frac{e^{-j\omega_i}}{1+e^{-j\omega_i}}} \right|, k \in \{1,2,3,4\}. \tag{5}$$

Requiring only four parallel multiplier blocks, the proposed differential-form VLSI architecture is significantly less complex than the direct-form version [15]. Moreover, the proposed 4-D differential-form architecture achieves a lower multiplier count by replacing the 4-D delay elements having $\mathbf{z}$-transforms of the form z_i^{-1}, $i \in \{s,t,u,v\}$, with 4-D differential operators having $\mathbf{z}$-transforms:

$$Y'_{ik}(z_i) = \frac{z_i^{-1}}{1 + z_i^{-1}} \cdot Y_k(\mathbf{z}),. \tag{6}$$

The differentiators described above are realized in the 4-D spatial domain using the following difference equation:

$$\begin{aligned} y'_{sk}[n_s, n_t, n_u, n_v] = \; & y_k[n_s - 1, n_t, n_u, n_v] \\ & - y'_{sk}[n_s - 1, n_t, n_u, n_v], \end{aligned} \tag{7}$$

$y'_{tk}[n_s, n_t, n_u, n_v], y'_{uk}[n_s, n_t, n_u, n_v], y'_{vk}[n_s, n_t, n_u, n_v]$ are easily generalized from (7), where $y'_{ik}[n_s, n_t, n_u, n_v]$ is the inverse $\mathbf{z}$-transforms of $Y'_{ik}(z_i)$.

3.1 4-D differential-form depth enhancers

The basic building block of 4-D light field filtering applications is the frequency-hyperplanar filter [6]. In recent work, it was demonstrated that two 2-D frequency-planar filters having Laplace transfer functions given as, where $(p,q) = (t,v)$ if $k = 1$, $(p,q) = (s,u)$, if $k = 2$,

$$H_k(\mathbf{s}) = H_k(s_p, s_q) = \frac{R_k}{R_k + L_p s_p + L_q s_q} = \frac{Y_k(s_p, s_q)}{W_k(s_p, s_q)}. \tag{8}$$

Here, R_1, R_2 are the ohmic resistive terminations of a 2-D Ramamoorty and Bruton [18] LC ladder prototype network. After applying the bilinear transform to (8), the Z-domain equations are given as

$$Y_k(\mathbf{z}) = W_k(\mathbf{z}) - Y_k(\mathbf{z})\alpha_p \frac{z_p^{-1}}{1 + z_p^{-1}} - Y_k(\mathbf{z})\alpha_q \frac{z_q^{-1}}{1 + z_q^{-1}}, \tag{9}$$

The difference equations obtained from (9) are implemented as depicted in Fig. 2.

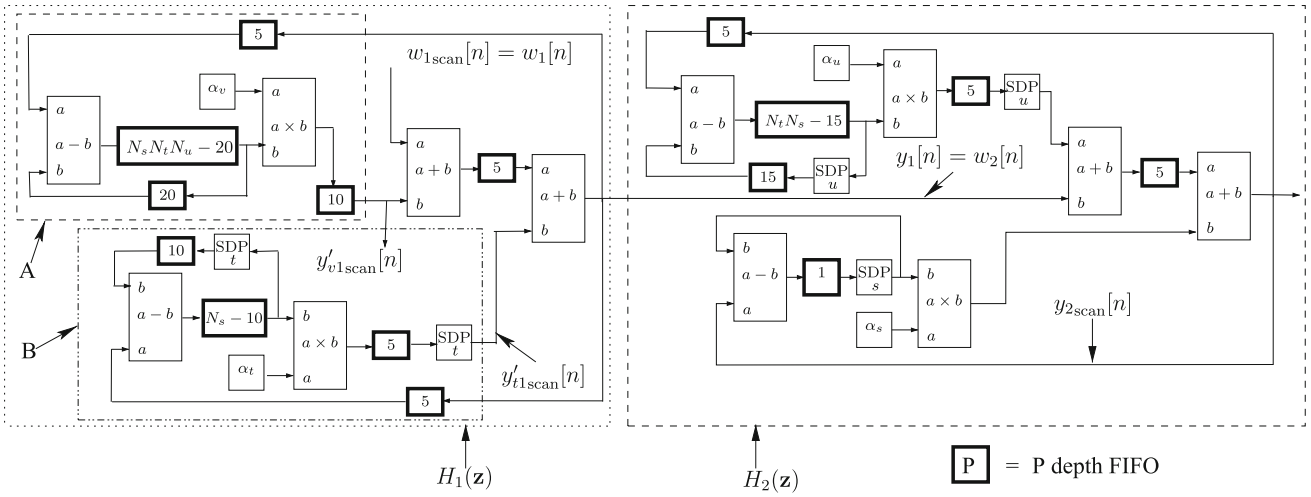

Fig. 2 The implementation of the first two **filtering stages of the overall system**, i.e., $H_1(\mathbf{z})$ and $H_2(\mathbf{z})$, where $\mathbf{z} \equiv (z_s, z_t, z_u, z_v)$

3.2 Parameter selection for depth enhancement

The 4-D planar filter selectively passes scene elements based on their distance from the camera. Different depths are selected by appropriately orienting the planar passband. For a filter-passing objects at depth p_z and d corresponding to the plane separation in the two-plane parameterization of the light field, the required passband plane is given as

$$\begin{bmatrix} 1 & 0 & 1-d/p_z & 0 \\ 0 & 1 & 0 & 1-d/p_z \end{bmatrix} \times \begin{bmatrix} \Omega_s \\ \Omega_t \\ \Omega_u \\ \Omega_v \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}, \tag{10}$$

where Ω_s, Ω_t, Ω_u, and Ω_v are continuous frequencies. Note that the plane (10) decomposes naturally into two hyperplanes with normals given by the rows of the left-hand matrix. Implementing the planar filter is therefore easily accomplished by cascading two hyperplanes with appropriately selected orientations. The continuous domain hyperplanar filter described in (3) has a normal given by $\mathbf{L}$. As a result, the construction of the planar filter reduces to selecting two hyperplane normals:

$$\begin{aligned} \mathbf{L}_1 &= \frac{1}{\sqrt{1+(1-d/p_z)^2}} \times \begin{bmatrix} 1 \\ 0 \\ 1-d/p_z \\ 0 \end{bmatrix}, \\ \mathbf{L}_2 &= \frac{1}{\sqrt{1+(1-d/p_z)^2}} \times \begin{bmatrix} 0 \\ 1 \\ 0 \\ 1-d/p_z \end{bmatrix}. \end{aligned} \tag{11}$$

In order to operate in a real-world system with nonuniform sample rates F_s, F_t, F_u, and F_v in dimensions s, t, u, and v, respectively, vector $\mathbf{L}$ must be modified as

$$\mathbf{L}_{\text{adjusted}} = \frac{\begin{bmatrix} L_s F_s & L_t F_t & L_u F_u & L_v F_v \end{bmatrix}^\top}{\left\| \begin{bmatrix} L_s F_s & L_t F_t & L_u F_u & L_v F_v \end{bmatrix}^\top \right\|}. \tag{12}$$

Example values for rate-adjusted inductors can be calculated for each of the four filtering stages and are depicted in Table 1.

3.3 Light field sampling

For a fixed aperture (n_s, n_t), we define the following 1-D discrete signal:

$$\tilde{w}[m; n_s, n_t] = w\left(\begin{bmatrix} n_s & n_t & n_u & n_v \end{bmatrix}^\top\right), \tag{13}$$

where $m = N_u n_v + n_u$.

These $N_s N_t$ raster-scanned sequences are concatenated by first up-sampling by factor $N_s N_t$ with zero padding and then folding. Therefore, we obtain the final time-interleaved sequence $w_{1\text{scan}}[n] = \tilde{w}[m; n_s, n_t]$, for $n = m + N_s N_t \times (N_s n_t + n_s)$. The resulting raster-scanned 4-D sequence $w_{1\text{scan}}[n]$ is submitted to filtering using the proposed architecture. These operations yield the filtered 4-D output raster-scanned sequence denoted by $y_{4\text{scan}}[n] \equiv y[\mathbf{n}]$.

3.4 Four-step zero-phase filtering

The proposed raster-scanned 4-D filter architecture is comprised of four steps: $y_k[\mathbf{n}] = h_k[\mathbf{n}] \overset{4\text{-D}}{*} w_k[\mathbf{n}]$, $k = 1, 2, 3, 4$, where discrete functions $w_k[\cdot]$, $y_k[\cdot]$, and $h_k[\cdot]$ correspond to the input, output, and impulse response of each frequency-hyperplanar digital filter. Note that $w_1[\mathbf{n}] = w[\mathbf{n}]$ and $y_4[\mathbf{n}] = y[\mathbf{n}]$. Linear 4-D convolution is achieved by iteratively calculating the 4-D difference equations which can be found via the inverse $\mathbf{z}$-transform of (4).

This four-step filtering process is conceptually broken into two stages. The first stage consists of $H_1(\mathbf{z})$ and $H_2(\mathbf{z})$, which

Table 1 Inductor and selectivity values for the two example designs (discussed in Sect. 4)

	Example 1				Example 2			
	$H_1(z)$	$H_2(z)$	$H_3(z)$	$H_4(z)$	$H_1(z)$	$H_2(z)$	$H_3(z)$	$H_4(z)$
L_s	0	0.9487	0	0.9487	0	0.5922	0	0.5922
L_t	0.9487	0	0.9487	0	0.5922	0	0.5922	0
L_u	0	0.3162	0	0.3162	0	0.8058	0	0.8058
L_v	0.3162	0	0.3162	0	0.8058	0	0.8058	0
R_0	0.05	0.05	0.05	0.05	0.02	0.02	0.02	0.02

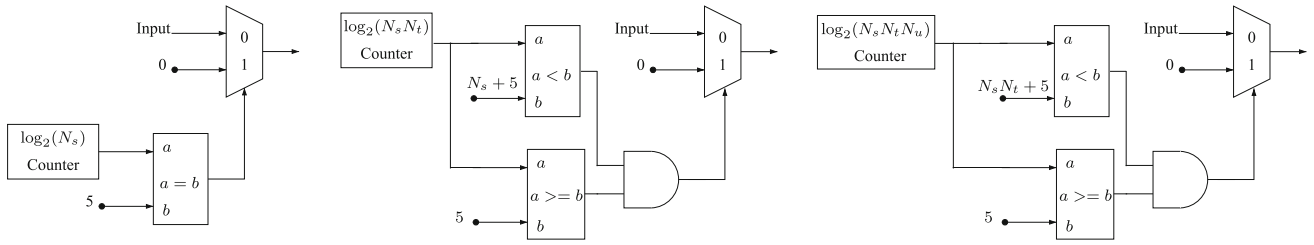

Fig. 3 Required spatial delay processors: **a** s-wise SDP (SDP s), **b** t-wise SDP (SDP t), and **c** u-wise SDP (SDP u)

accomplishes depth selectivity. The second stage consists of $H_3(\mathbf{z})$ and $H_4(\mathbf{z})$, which increases selectivity and also enforces zero-phase filtering. Because all four independent variables are spatial in nature, zero-phase filtering is accomplished in the second stage by flipping the signals along all dimensions s, u and t, v and re-filtering using the same hardware architecture. This procedure leads to no phase distortion from 4-D IIR filters such that the following expression holds true:

$$|H_1(e^{j\omega_s}, e^{j\omega_u}) \cdot H_2(e^{j\omega_t}, e^{j\omega_v})| = |H_3(e^{-j\omega_s}, e^{-j\omega_u}) \cdot H_4(e^{-j\omega_t}, e^{-j\omega_v})|, \quad (14)$$

and

$$\angle|H_1(e^{j\omega_s}, e^{j\omega_u}) \cdot H_2(e^{j\omega_t}, e^{j\omega_v})| + \angle|H_3(e^{-j\omega_s}, e^{-j\omega_u}) \cdot H_4(e^{-j\omega_t}, e^{-j\omega_v})| = 0, \quad (15)$$

where $\angle$ returns the phase angle of its argument.

4 FPGA-based hardware implementation

4.1 Hardware architecture

The proposed architecture employs a 4-D raster-scanned input signal. The architecture directly operates on the raster-scanned signal to compute the difference equations associated with (4). This corresponds to two parallel differentiators, not four, because two of the differentiator values $\alpha_i, i \in \{s, t, u, v\}$ are zero.

The output of the parallel differentiators is denoted by $y'_{ik\,\mathrm{scan}}[n]$, $i \in \{s, t, u, v\}$, $k \in \{1, 2, 3, 4\}$ which corresponds to the signals $y'_{ik}[n_s, n_t, n_u, n_v]$ described in (7) after the application of the raster scanning procedure described in Sect. 3.3.

The final filtered output computation is given by:

$$y_{k\,\mathrm{scan}}[n] = w_{k\,\mathrm{scan}}[n] - \sum_{i\in\{s,t,u,v\}} \alpha_i\, \phi_i[n]\, y'_{ik\,\mathrm{scan}}[n], \quad (16)$$

where $k \in \{1, 2, 3, 4\}$, $\phi_s[n]$ and $\phi_t[n]$ are zero initial conditions (ZICs) applied along columns and rows in the s, t plane, and $\phi_u[n]$ and $\phi_v[n]$ are ZICs applied along columns and rows on the u, v plane, respectively.

The ZICs are achieved using a 4-D extension of the 3-D spatial delay processors (SDPs) proposed in [14]. The s-wise SDPs realize $\phi_s[n] = 0$, if $n_s = 0$, else $\phi_s[n] = 1$, and the t-wise SDPs realize $\phi_t[n] = 0$, if $n_t = 0$, else $\phi_t[n] = 1$. The ZICs for $\phi_u[n]$ and $\phi_v[n]$ are similarly defined for u-wise and v-wise recursions. These SDP processors are implemented in the differential-form filter architecture, depicted in Fig. 2 as SDP s, SDP t, and SDP u (Fig. 3).

Implementation of the required SDPs are shown in Fig. 3. As shown in Fig. 3a, when the counter value becomes five, it implies that $n_s = 0$. If the circuit was not pipelined, then counter value zero implies that $n_s = 0$. Therefore, a zero signal should be sent as output to satisfy the ZIC. Similarly, in Fig. 3b, when the counter values become less than $N_s + 5$ and more than or equal to 5, zeros should be injected because

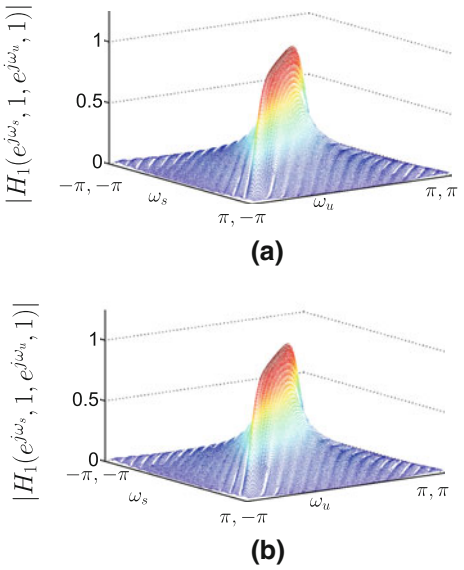

Fig. 4 **a** $|H_1(e^{j\omega_s}, 1, e^{j\omega_u}, 1)|$ of the ideal 2-D magnitude frequency response and **b** 2-D FFT of on-chip FPGA measurements. *Plots* are shown over $-\pi \leq \omega_s < \pi, -\pi \leq \omega_u < \pi$

that condition implies $n_t = 0$. The operation of the SDP circuit shown in Fig. 3c follows a similar rationale.

The complete implementation of the filter architecture described in (9) is detailed in Fig. 2. The difference equation of (9) (when $(p, q) = (t, v)$) can be calculated for raster-scanned 4-D signals using (16) and is given by

$$y_1[n] = w_1[n] - \alpha_t \phi_t[n] y'_{t1\,\mathrm{scan}}[n] - \alpha_v \phi_v[n] y'_{v1\,\mathrm{scan}}[n]. \tag{17}$$

Here $y_1[n] = y_{1\,\mathrm{scan}}[n] \leftrightarrow Y_1(z)$ and $w_1[n] = w_{1\,\mathrm{scan}}[n] \leftrightarrow W_1(z)$. The parallel differentiators $y'_{t1\,\mathrm{scan}}[n]$, $y'_{v1\,\mathrm{scan}}[n]$ are given as

$$y'_{t1\,\mathrm{scan}}[n] = y'_{t1}[n_s, n_t, n_u, n_v] = y_1[n_s, n_t - 1, n_u, n_v] - y'_{t1}[n_s, n_t - 1, n_u, n_v], \tag{18}$$

The $y'_{v1\,\mathrm{scan}}[n]$ calculation is omitted for brevity. The unit delay in n_t is mapped to an N_s-deep FIFO. The SDP t circuit in block B of Fig. 2 and the other mappings follow from the above difference equation. The unit delay in n_v is mapped to an $N_s N_t N_u$-deep FIFO.

This method can also be applied to calculate the difference equation of (9) when $(p, q) = (s, u)$. The inverse **z**-transform of (9) gives: $y_{2\,\mathrm{scan}}[n] \leftrightarrow Y_2(z)$. Also, we have the following mapping: $w_2[n] = w_{2\,\mathrm{scan}}[n] \leftrightarrow W_2(z)$. Furthermore, from Fig. 2, it is clear that 2 multipliers and 4 adder/subtracter blocks are required for the realization of a single 4-D IIR filter block.

4.2 Differential-form filter implementation and test

We employ reconfigurable logic platforms for the rapid prototyping of proposed low-complexity 4-D filter hardware. The proposed design achieves 4-D IIR filtering using a partially separable approach, where two 2-D filters operating on the s, u and t, v dimensions are cascaded to obtain the required 4-D IIR filter. The design was implemented in a Xilinx ML605 board from Avnet populated with a single Xilinx Virtex-6 xc6vsx315t-3ff1156 FPGA device. The digital logic required for the simulation was supported by the hardware co-simulation provided by the ML605 board. The 2-D unit impulse response of each sub-filter $H_i(\mathbf{z}), i = 1, 2, 3, 4$, connected in cascade in the total realization was measured on-chip using stepped hardware co-simulation. The measured 2-D magnitude frequency responses are obtained and compared against theoretical responses as illustrated in Fig. 4.

4.3 Internal precision and resource consumption

The fixed-point arithmetic parameters are in Table 4, where W and D refer to input word size and binary point position, respectively. Variables W_{in} and D_{in} define fixed-point arithmetic precision. Further, W_{out} and D_{out} define the precision of the filter outputs, while W_{mul} and D_{mul} define the coefficient precision. If the VLSI hardware requirements for W-bit multipliers and adders are γ_{M} and $\gamma_{\mathrm{A/S}}$, respectively, then the total VLSI resource consumption of a circuit is approximated by $\gamma_T \approx 2 \cdot \gamma_{\mathrm{M}} + 4 \cdot \gamma_{\mathrm{A/S}} + K$, where K is the VLSI resource requirements for the ZIC circuits.

Table 3 gives a comparison between the number of multiplier and adder resources used for direct form II [15] and differential form. From Table 3, the resource usage of the differential-form filter is significantly lower than the direct-form architecture. That is, differential-form architectures reduce the multiplier complexity from 15 down to 2; adders reduced from 19 down to 4. It can be further seen from Table 2 that the FPGA resource consumption in the differential-form filter is significantly lower when compared with the direct form II filter.

Table 2 was chosen from the minimum operating frequency of $H_1(\mathbf{z})$ and $H_2(\mathbf{z})$ from Table 4. The critical path delay (CPD) of the $H_1(\mathbf{z})$ circuit implementation is given by $T_{\mathrm{CPD1}} \approx T_{\mathrm{A/S}} + T_{\mathrm{MUX}}$ where T_{M}, $T_{\mathrm{A/S}}$, and T_{MUX} are the propagation delays of a parallel multiplier, adder/subtractor, and two-input W-bit multiplexer, respectively.

The critical path delay (CPD) of the $H_2(\mathbf{z})$ circuit implementation is given by $T_{\mathrm{CPD2}} \approx T_{\mathrm{M}} + 2T_{\mathrm{A/S}} + T_{\mathrm{MUX}}$. The maximum clock frequency for $H_1(\mathbf{z})$ circuit implementation is $F_{\mathrm{clock}} = 1/T_{\mathrm{CPD1}}$ and similarly for $H_2(\mathbf{z})$ circuit implementation. It is recalled from [14] that multiplexers are

Table 2 Performance comparison of the differential-form and direct-form II filters

Parameter	Differential-form	Direct form II
W_{in}	19	19
D_{in}	6	6
W_{mul}	18	18
D_{mul}	12	12
N_s	32	32
N_t	32	32
N_u	256	256
N_v	256	256
W_{out}	26	26
D_{out}	6	6
Maximum frequency (MHz)	55.35	68.63
Number of occupied slices	680	2,260
Number of LUT flip–flop pairs	2,608	8,545
Number of slice LUTs	2,596	8,388
Number of DSP48s	6	36
Number of bonded IOBs	44	78
Number used as BUFGs	1	1

Table 3 Multiplier and adder resource usage in different filter implementations

Architecture	Number of adders	Number of multipliers
Direct form II	19	15
Differential form	4	2

required when realizing the SDP circuits for obtaining the ZICs in the recursions along the four dimensions.

4.4 Estimated throughput

The real-time throughput is $8 \cdot F_{\text{clock}}$ fixed-point multiplications and $16 \cdot F_{\text{clock}}$ additions/subtractions per second, corresponding to a light field process rate given by

$$F = \frac{\min\left(F_1, F_2, F_3, F_4\right)}{4N_s N_t N_u N_v} \text{ Hz}. \tag{19}$$

where F_1, F_2, F_3, F_4 correspond to the operating frequencies of the four filter stages H_1, H_2, H_3, H_4 shown in Fig. 1. For a light field having dimensions $N_s N_t N_u N_v$, the architecture completes the filtering operation in $T = 1/F$ s. Considering $N_s = N_t = 16$, $N_u = N_v = 128$, this period corresponds to 0.26 s for a maximum operating frequency of 65.1 MHz. The video frame rate achievable is therefore about four 4-D light fields per second. However, the achieved frame rate is for our proof-of-concept design, which can clearly be increased with improvements in FPGA technology or by the use of application-specific integrated circuits as shown in the ASIC realization of the circuit described below. A second design example was also pursued, yielding similar results as summarized in Table 4. More details of both examples follow in the next section.

Table 4 Design parameters and measured results for $H_k(\mathbf{z})$, $k = 1, 2, 3, 4$ of Examples 1 and 2

Parameter	Measured value Example 1				Measured value Example 2			
	$H_1(z)$	$H_2(z)$	$H_3(z)$	$H_4(z)$	$H_1(z)$	$H_2(z)$	$H_3(z)$	$H_4(z)$
W_{in}	9	19	23	27	12	19	26	26
D_{in}	0	6	6	6	11	6	6	6
W_{mul}	18	18	18	18	18	18	18	18
D_{mul}	12	12	12	12	12	12	12	12
N_s	16	16	16	16	32	32	32	32
N_t	16	16	16	16	32	32	32	32
N_u	128	128	128	128	256	256	256	256
N_v	128	128	128	128	256	256	256	256
W_{out}	19	23	27	32	19	26	26	33
D_{out}	6	6	6	6	6	6	6	6
Maximum frequency (MHz)	158.71	66.84	133.5	65.1	154.75	55.35	155.98	52.1
Number of occupied slices	256	544	411	716	124	680	136	877
Number of LUT flip–flop pairs	885	2,091	1,491	2,763	472	2,608	490	3,370
Number of slice LUTs	885	2,091	1,491	2,763	472	2,596	490	3,359
Number of DSP48s	6	6	12	12	2	6	2	8
Number of bonded IOBs	193	130	334	178	87	44	103	58
Number used as BUFGs	1	1	1	1	1	1	1	1

5 Verification of video surveillance

5.1 Example 1

Verification of Example 1 is achieved using a sample gantry-captured light field obtained from the *Stanford light field archive* [20]. The example provided is "Toy Humvee and soldier behind dense foliage" (sample 1), which contains a view of a toy soldier behind a toy Humvee occluded by dense foilage (trees and shrubs) as shown in Fig. 5. The light field contains a 16×16 array of images corresponding to a single frame in a light field video sequence.

The objective is to process this video frame to attenuate the dense foliage and observe the soldier and Humvee. Any single-camera view is highly occluded by foliage. However, each view in the light field contains partial components of the desired scene despite the high degree of occlusion from dense foliage. Adapting the filters utilized here to a particular light field geometry is simply accomplished by adjusting the filter coefficients using (12). An important limitation is that as the number of samples in a dimension decrease, the ability of the filter to discriminate depths also decreases. As a general rule of thumb, there should be no fewer than 10 samples in any dimension. A higher image resolution will not necessarily yield proportionally superior performance.

Figure 5a shows a sample image of the Humvee without occlusions, Fig. 5b depicts a 128×128 image from one of the apertures of the input light field $w[\mathbf{n}]$, while Fig. 5c depicts the filtered light field, clearly showing enhancement of the soldier and Humvee and attenuation of the occluding dense foliage. For comparison, the ideal depth-filtered Humvee image is shown in Fig. 5d where the results are obtained by filtering the light field using 64-bit floating point arithmetic in the MATLAB environment. Subjectively, the filtered light field results of the Humvee clearly show its headlights and vehicle shape which are completely covered in individual apertures from the input light field video sequence.

5.2 Example 2

In a second example, we employ monochrome imagery. We consider a beverage coaster mounted at a depth of 22 cm from a background poster showing a supernova (provided by the Dominion Radio Astrophysical Observatory in Penticton, BC, Canada). The light field was obtained at the University of Calgary using an experimental gantry [6]. The dimensions of the light field are $N_s = N_t = 32$ and $N_u = N_v = 256$. A digital camera at depth 66 cm having image resolution 256×256 mounted on a moving platform in the s, t plane is used to capture the 32×32 image matrix.

A sample image of the light field is shown in Fig. 6a, while the output of $H_4(\mathbf{z})$ is shown in Fig. 6b. Clearly, the occluder (a beverage coaster) was almost entirely removed, while the area behind it which was previously occluded is now exposed showing significant detail [6]. In essence, we are "looking behind" the occluder.

5.3 Performance of the depth-filtered Humvee images

Let the image intensity matrices be denoted by I_{ref}, I_{in}, I_{outM}, I_{outH} for the un-occluded Humvee image, occluded Humvee image, depth-filtered Humvee obtained by difference equation using Matlab, and depth-filtered Humvee obtained by hardware simulation. The Humvee images were taken by resizing the images to a resolution of $(x \times y) \in (300, 300)$, converted to gray scale, and then normalized. Subsequently, the auto/cross-correlation was calculated according to:

$$X_{(g,h)}(i, j) = \sum_{n=0}^{(x-1)} \sum_{p=0}^{(y-1)} I_g(n, p) I_h(n + i, p + j), \qquad (20)$$

where $(g, h) \in \{(I_{\text{ref}}, I_{\text{ref}}), (I_{\text{ref}}, I_{\text{in}}), (I_{\text{ref}}, I_{\text{outH}}), (I_{\text{ref}}, I_{\text{outM}})\}$, $0 \le i < (2x - 1)$, and $0 \le j < (2y - 1)$.

For comparison purposes, we define a performance metric $\lambda_{(a,b,c,d)}$:

$$\lambda_{(a,b,c,d)} = 10 \log_{10} \left(\frac{\max \left(X_{(a,b)}(i, j) \right)}{\max \left(X_{(c,d)}(i, j) \right)} \right), \qquad (21)$$

where $(a, b, c, d) \in \{(I_{\text{ref}}, I_{\text{in}}, I_{\text{ref}}, I_{\text{ref}}), (I_{\text{ref}}, I_{\text{outH}}, I_{\text{ref}}, I_{\text{ref}}), (I_{\text{ref}}, I_{\text{outM}}, I_{\text{ref}}, I_{\text{ref}})\}$. The computed values for $\lambda_{(I_{\text{ref}}, I_{\text{in}}, I_{\text{ref}}, I_{\text{ref}})}$, $\lambda_{I_{\text{ref}}, I_{\text{outH}}, I_{\text{ref}}, I_{\text{ref}}}$, $\lambda_{(I_{\text{ref}}, I_{\text{outM}}, I_{\text{ref}}, I_{\text{ref}})}$ were -8.98, -0.014, and -0.009 dB, respectively. An improvement of 8.966 dB can be observed in the hardware-filtered Humvee image and an improvement of 8.971 dB can be observed in the Humvee image filtered by the difference equation implemented in Matlab. From the results, we can clearly see that the filtered images are much closer to the original Humvee toy image.

5.4 Hardware parameter selection

In Example 1, the RGB channels of input images are separately applied to three 4-D IIR filters, using 8 bits per color depth. The input word size W_{in} should be within a value of $N = 8$ bits with the binary point at zero ($D_{\text{in}} = 0$). The fixed-point precision levels are summarized in Table 4. In Example 2, input signals are monochrome and therefore processed using a single filter. To obtain the monochrome light field, three RGB color channels were summed, and the resulting image was normalized to unity and quantized to 12-bits of precision. The input system word size was set to 12 bits with the binary bit position at $D_{\text{in}} = 11$. The fixed-point precision is given in Table 4. The input word length for

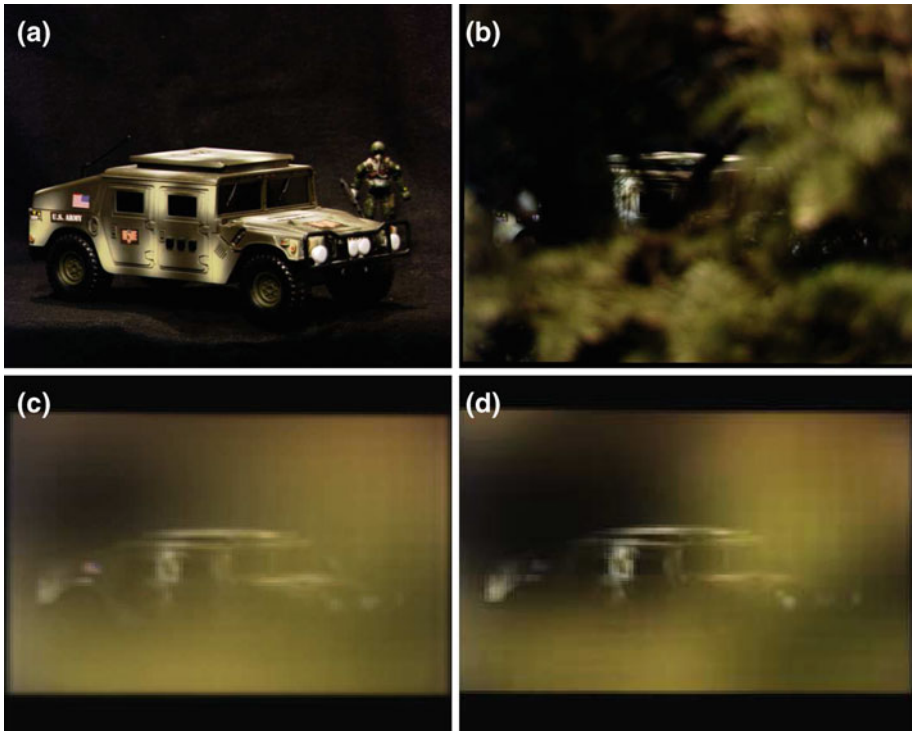

Fig. 5 **a** Sample image of the Humvee without occlusions; **b** one of the 128 × 128 images of w [**n**] of the light field input signal with occlusions due to trees/shrubs; **c** depth-filtered Humvee obtained from the FPGA-based hardware implementation; and **d** Ideal depth-filtered Humvee image obtained from MATLAB

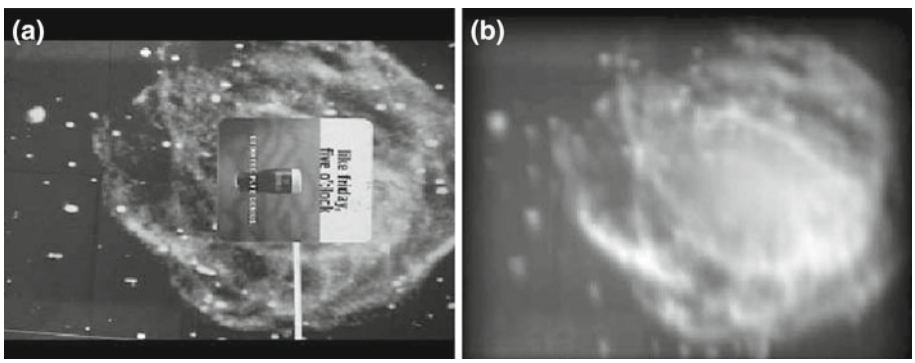

Fig. 6 **a** A sample of the light field and **b** the hardware depth-filtered output

the four filters is increased because of the growth in system word size and signal-to-noise ratio as the filtering operations progress along the cascade of 4-D IIR filtering stages. Furthermore, the maximum frequency of $H_3(\mathbf{z})$ is higher than $H_2(\mathbf{z})$. This is nonintuitive given the higher word lengths of $H_3(\mathbf{z})$. The increased processing speed of $H_3(\mathbf{z})$ is due to the reduced complexity of the SDP circuitry, which dominates critical path delay over system word length in this example.

5.5 ASIC synthesis using 45 nm CMOS

The two design examples were synthesized for application-specific integrated circuits (ASIC) using the Cadence Encounter RTL Compiler for 45 nm CMOS technology. Table 5 displays the area, power, and speed results for the ASIC implementation at typical operating conditions with a supply voltage of 1.1 V and operating temperature of 27 °C. The area and power consumption for $H_1(\mathbf{z})$ and $H_3(\mathbf{z})$ are lower than the values obtained for $H_2(\mathbf{z})$ and $H_4(\mathbf{z})$. This is due to the large delay buffer of size $N_s N_t N_u - 20$ (Fig. 2) being implemented in Matlab rather than being synthesized in the ASIC synthesis flow. For 45 nm CMOS, the video frame rate achievable for Example 1 at a maximum operating frequency of 442.86 MHz was 26.39 Hz; for Example 2, at a maximum operating frequency of 453.92 MHz, it was 1.69 Hz, this reduction in value was due to the larger light field size.

6 Conclusion

In mission-critical applications, unobstructed visibility of the area, objects, and/or persons of interest is almost never guaranteed due to clutter and distractors. We propose the

Table 5 Speed of operation, power consumption, and area utilization for an ASIC implementation (synthesis only) for Examples 1 and 2 using 45 nm CMOS Technology

Architecture	Area (mm^2)	Static power (mW)	Dynamic power (mW)	Total power (mW)	Max. Freq. (MHz)
$H_1(z)$ Example 1	0.077	0.607	118.693	119.301	796.81
$H_2(z)$ Example 1	0.650	5.330	928.238	933.569	450.04
$H_3(z)$ Example 1	0.133	1.000	192.221	193.222	727.27
$H_4(z)$ Example 1	0.879	7.183	1,255.400	1,262.58	442.86
$H_1(z)$ Example 2	0.0560	0.443	89.830	90.275	739.1
$H_2(z)$ Example 2	1.322	10.965	1,934.88	1,945.85	465.54
$H_3(z)$ Example 2	0.059	0.462	93.510	93.972	757.57
$H_4(z)$ Example 2	1.725	14.312	2,520.489	2,534.801	453.92

application of plenoptic cameras in such scenarios, enabling real-time or post-capture filtering of the incoming light field. Depth selectivity and occluder removal is achieved with a novel differential-form 4-D IIR depth-enhancing filter. The proposed low-complexity differential-form architecture requires significantly less hardware when compared to competing filters. The proposed design effectively requires only 2 multipliers per 4-D IIR filter, compared with the 15 multipliers required by recently proposed [15] filters. An FPGA-based prototype implementation of the low-complexity architecture was produced and applied to light fields from the Stanford Light Field Archive and the University of Calgary. The filter was observed to operate correctly, clearly demonstrating how the proposed technology enables the removal of occluding objects in video sequences, yielding unobstructed views of the objects of interest and delivers 26 frames/s for light fields of size $16 \times 16 \times 128 \times 128$.

References

1. Adelson, E.H., Bergen, J.R.: The plenoptic function and the elements of early vision. Comput. Models Vis. Process. **1**, 3–20 (1991)
2. Agathoklis, P., Bruton, L.T.: Practical-BIBO stability of N-dimensional discrete systems. Electron. Circuits Sys. IEEE Proc. G **130**(6), 236–242 (1983)
3. Bertschmann, R.K., Bartley, N.R., Bruton, L.T.: A 3-D integrator-differentiator double-loop (IDD) filter for raster-scan video processing. In: IEEE international symposium on circuits and systems, ISCAS'95, vol. 1, pp. 471–473 (1995)
4. Bruton, L.T., Bartley, N.R.: Three-dimensional image processing using the concept of network resonance. IEEE Trans. Circuits Sys. **32**(7), 664–672 (1985)
5. Dansereau, D.G., Bruton, L.T.: A 4D frequency-planar IIR filter and its application to light field processing. In: Circuits and systems, 2003. ISCAS '03, vol. 4, pp. 476–479 (2003)
6. Dansereau, D.G., Bruton, L.T.: A 4-D dual-fan filter bank for depth filtering in light fields. IEEE Trans. Signal Process. **55**(2), 542–549 (2007)
7. Dansereau, D.G., Mahon, I., Pizarro, O., Williams, S.B.: Plenoptic flow: Closed-form visual odometry for light field cameras. In: Proceedings, 2011 IEEE/RSJ international conference on intelligent robots and systems (IROS), pp. 4455–4462. IEEE (2011)
8. Harris, M.: Focusing on everything: light field cameras promise an imaging revolution. IEEE Spectr. **49**, 44–50 (2012)
9. Jarvis, R.A.: A perspective on range finding techniques for computer vision. Pattern Anal. Mach. Intell. IEEE Trans. PAMI **5**, 122–139 (1983)
10. Joshi, N., Avidan, S., Matusik, W., Kriegman, D.J.: Synthetic aperture tracking: tracking through occlusions. In: IEEE international conference on computer vision (ICCV), pp. 1–8. IEEE (2007)
11. Lai, S.H., Fu, C.W., Chang, S.: A generalized depth estimation algorithm with a single image. Pattern Anal. Mach. Intell. IEEE Trans. **14**, 405–411 (1992)
12. Levoy, M., Hanrahan, P.: Light field rendering. In: Proceedings of the 23rd annual conference on computer graphics and interactive techniques, pp. 31–42. ACM, New York (1996)
13. Lumsdaine, A., Georgiev, T.: Full resolution lightfield rendering. Technical report, Adobe Systems (2008)
14. Madanayake, A., Bruton, L.T.: A fully-multiplexed first-order frequency-planar module for fan, beam, and cone plane-wave filters. IEEE Trans. Circuits Sys. II Express Briefs **53**(8), 697–701 (2006)
15. Madanayake, A., Wimalagunaratne, R., Dansereau, D.G., Bruton, L.T.: Design and FPGA-implementation of 1st-order 4D IIR frequency-hyperplanar digital filters. In: 2011 IEEE 54th international midwest symposium on circuits and systems (MWSCAS), pp. 1–4 (2011)
16. Ng, R.: Fourier slice photography. ACM Trans. Graph. (TOG) **24**, 735–744 (2005)
17. Ng, R., Levoy, M., Brédif, M., Duval, G., Horowitz, M., Hanrahan, P.: Light field photography with a hand-held plenoptic camera. Computer Science Technical Report CSTR 2 (2005)
18. Ramamoorthy, P.A., Bruton, L.T.: Design of stable 2-dimensional discrete recursive filters. Electron. Lett. **12**(25), 659–660 (1976)
19. Smith, B.M., Zhang, L., Jin, H., Agarwala, A.: Light field video stabilization. In: 2009 IEEE 12th international conference on computer vision, pp. 341–348. IEEE (2010)
20. The (New) Stanford Light Field Archive (2012) http://lightfield.stanford.edu/
21. Vaish, V., Levoy, M., Szeliski, R., Zitnick, C., Kang, S.: Reconstructing occluded surfaces using synthetic apertures: Stereo, focus and robust measures. In: 2006 IEEE computer society conference on computer vision and pattern recognition, vol. 2, pp. 2331–2338 (2006)
22. Wilburn, B., Joshi, N., Vaish, V., Talvala, E.V., Antunez, E., Barth, A., Adams, A., Horowitz, M., Levoy, M.: High performance imaging using large camera arrays. ACM Trans. Graph. (TOG) **24**, 765–776 (2005)