Optical Flow Estimation Using Wavelet Motion Model

[1,3]Yu-Te Wu [1]Takeo Kanade [2]Jeffrey Cohn [3]Ching-Chung Li

[1]The Robotics Institute
Carnegie Mellon University
Pittsburgh, PA 15213

[2]Dept. of Psychology and Psychiatry
University of Pittsburgh
Pittsburgh, PA 15260

[3]Dept. of Electrical Engr.
University of Pittsburgh
Pittsburgh, PA 15260

Abstract

A motion estimation algorithm using wavelet approximation as an optical flow model has been developed to estimate accurate dense optical flow from an image sequence. This wavelet motion model is particularly useful in estimating optical flows with large displacement. Traditional pyramid methods which use the coarse-to-fine image pyramid by image burring in estimating optical flow often produce incorrect results when the coarse-level estimates contain large errors that cannot be corrected at the subsequent finer levels. This happens when regions of low texture become flat or certain patterns result in spatial aliasing due to image blurring. Our method, in contrast, uses large-to-small full-resolution regions without blurring images, and simultaneously optimizes the coarser and finer parts of optical flow so that the large and small motion can be estimated correctly. We compare results obtained by using our method with those obtained by using one of the leading optical flow methods, the Szeliski pyramid spline-based method. The experiments include cases of small displacement (less than 4 pixels under 128× 128 image size or equivalent displacement under other image sizes), and those of large displacement (10 pixels). While both methods produce comparable results when the displacements are small, our method outperforms pyramid spline-based method when the displacements are large.

1 Introduction

This paper presents a method to estimate optical flow using coarse-to-fine wavelet representation, newly presented by Cai and Wang [5], as a motion model. The wavelet motion model represents motion vectors by a linear combination of hierarchical basis functions. The coarser-scale basis function has larger support while the finer-scale basis function has smaller support. Corresponding to these variably sized supports, large-to-small full-resolution regions are used in image matching. In addition, the associated coefficients to be estimated have global and local influences in constructing the motion vectors.

The major feature of Cai-Wang basis functions is to directly transform any function into wavelet coefficients from coarse to fine scale. This differs from the conventional usage of wavelet transform which is carried out from fine-to-coarse for decomposition and then coarse-to-fine for reconstruction. This feature allows us to estimate the coefficients starting from coarsest scale by using the largest full resolution patches. This is particularly useful in estimating the optical flow due to large motion. At each iteration, we reconstruct the motion vectors from the first to the second image based on the estimated coefficients, and use them to warp the first image. As finer-scale coefficients are estimated by minimizing the sum of squared intensity differences between the warped image and the second image, we obtain more accurate results.

To handle large displacement, a number of coarse-to-fine hierarchical methods [2, 4, 1] have been developed. The pyramid methods which use coarse-to-fine blurred images sequentially in estimating optical flow often produce incorrect results when large errors occurring in coarser estimates cannot be subsequently corrected in the finer estimates. This happens when regions of low texture become flat or certain patterns result in spatial aliasing due to image blurring. The reason that our method could cope with this difficulty is that our representation hierarchy is different from the typical image coarse-to-fine hierarchy. Our hierarchy is built along motion resolution level, not image resolution level. In addition, we use large-to-small full resolution regions, rather than blurred images, in estimating the coefficients from coarse to fine scales.

2 Mathematical Background

Let $I \stackrel{\text{def}}{=} [0, L]$ denote any finite interval, where $L \geq 4$ is a positive integer, and $H^2(I)$ denote Sobolev space which contains all continuous functions with finite energy norm up to the second derivative, respectively, i.e.,

$$H^2(I) \stackrel{\text{def}}{=} \{f(x), x \in I | \|f^{(i)}\|_2 = \int_I |f^{(i)}(x)|^2 dx \ < \infty, i = 0, 1, 2.\}$$

It can be verified that $H^2(I)$ is a Hilbert space with the inner product, $< f,\ g >= \int_I f^{(2)}(x) g^{(2)}(x) dx$. Hence $|||f||| = < f,\ f >^{\frac{1}{2}}$ provides a norm for $H^2(I)$. In order to develop a coarse-to-fine wavelet representation in $H^2(I)$, Cai and Wang used the fourth-order B-spline $\phi(x)$ (Figure 1.(a))

$$\phi(x) \stackrel{\text{def}}{=} \frac{1}{6} \sum_{j=0}^{4} \binom{4}{j} (-1)^j (x - j)_+^3 \in H^2(I) \qquad (1)$$

as the scaling function, where $x_+^n = x^n$ if $x \geq 0$, and $x_+^n = 0$ if $x < 0$. Using the scaling function, they

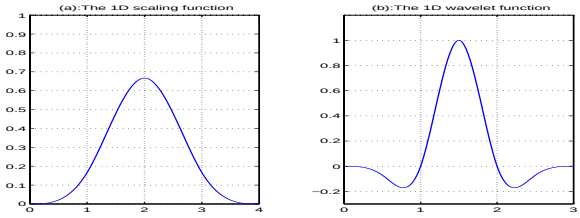

Figure 1: The 1D scaling and wavelet functions.

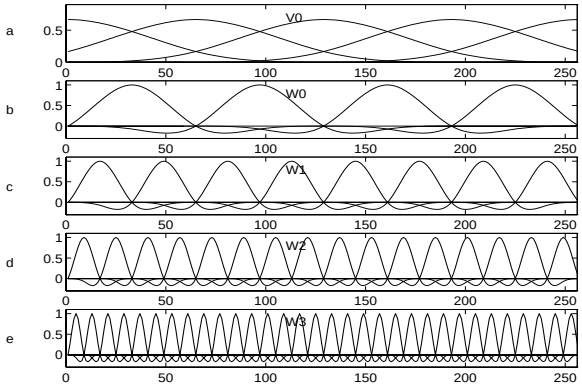

Figure 2: The translation and dilation of basis functions at different resolution scales

constructed the wavelet (Figure 1.(b))

$$\psi(x) = \frac{-3}{7}\phi(2x) + \frac{12}{7}\phi(2x-1) - \frac{3}{7}\phi(2x-2) \qquad (2)$$

The supports of $\phi(x)$ and $\psi(x)$ are $[0,4]$ and $[0,3]$, respectively, i.e., the values of $\phi(x)$ (or $\psi(x)$) are zeros outside $[0,4]$ (or $[0,3]$). The dilation and translation of $\phi(x)$ and $\psi(x)$ are defined by

$$\begin{aligned} \phi_{j,k}(x) &= \phi(2^j x - k),\ j \geq 0,\ k = -2, \ldots, 2^j L - 2 \qquad (3) \\ \psi_{j,k}(x) &= \psi(2^j x - k),\ j \geq 0,\ k = -1, \ldots, 2^j L - 2 \end{aligned}$$

For each resolution level j, let V_j and W_j be the closure (under norm $|||f|||$) of linear span of $\{\phi_{j,k}(x),\ -2 \leq k \leq 2^j L - 2\}$ and $\{\psi_{j,k}(x) |\ k = -1, \ldots, 2^j L - 2\ \}$, respectively. Cai and Wang have shown that[5] any continuous function $d(x) \in H^2(I)$ can be approximated as closely as possible by a function in V_j for a sufficiently large j which has a unique orthogonal approximation

$$d(x) \approx d_{-1}(x) + d_0(x) + d_1(x) + \cdots d_j(x) \qquad (4)$$

where

$$d_{-1}(x) = \sum_{k=-2}^{L-2} c_{-1,k}\phi_{0,k}(x) \quad \in V_0 \qquad (5)$$

$$d_j(x) = \sum_{k=-1}^{2^j L-2} c_{j,k}\psi_{j,k}(x) \quad \in W_j \quad \text{for} \quad j \geq 0. \qquad (6)$$

Each component d_j is produced by the expansion of translated basis functions at different resolution levels where the coefficients $c_{j,k}$ are appropriately determined. Although the existence of approximation is proved for any continuous function in $H^2(I)$, it has been demonstrated in [5, 8] that it holds for any finite sampled function in the practical applications.

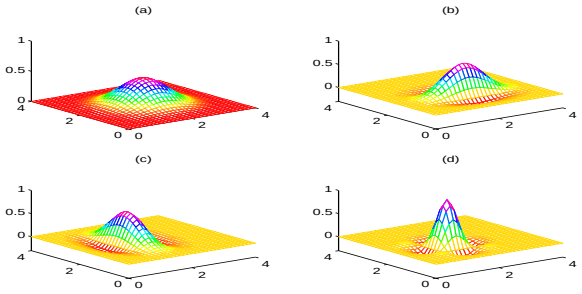

Figure 3: The two dimensional basis functions. (a) $\phi(x)\phi(y)$, (b) $\phi(x)\psi(y)$, (c) $\psi(x)\phi(y)$ and (d) $\psi(x)\psi(y)$.

Influence of coefficients $c_{j,k}$ can be global or local depending on the size of support of the corresponding basis function. To see this, let us assume $L = 4$ for simplicity such that the scaling functions $\phi(x-k)$ in the coarsest scale have support $[0,4]$ (except on the boundary), and the wavelet functions $\psi_{j,k}$ have (narrower) support $[0, \frac{3}{2^j}]$ (except near the boundary), $j \geq 0$. Let us also assume a finite sampled function defined on pixel domain $[0, 256]$. We map $[0,4]$ into $[0,256]$ so that the effects are interpreted on pixel unit. The perturbs of $c_{-1,-2}$, $c_{-1,-1}$, $c_{-1,0}$, $c_{-1,1}$ and $c_{-1,2}$ will affect the values $d_{-1}(x)$ globally within the domain $[0,127]$, $[0,191]$, $[0,256]$, $[65,256]$ and $[129,256]$, respectively (Figure 2). In the finer scales (levels) $j \geq 0$, wavelets have support $\frac{192}{2^j}$ pixels within the domain (and $\frac{128}{2^j}$ pixels on the boundaries), and there are total 2^{j+2} shifted wavelets overlapped across $[0,256]$. The perturbs of each wavelet coefficient $c_{j,k}$ will affect $d_j(x)$ locally only inside the domain $[\frac{64k}{2^j}, \frac{64(k+3)}{2^j}]$, and on the boundraries $[0, \frac{128}{2^j}]$ and $[256 - \frac{128}{2^j}, 256]$.

3 Wavelet-Based Flow Estimation

Suppose image $I_0(x,y)$ and $I_1(x,y)$ are related by horizontal and vertical displacement (flow or motion) vectors $u(x,y)$ and $v(x,y)$ under the *intensity constancy* constraint [3]. We wish to recover the maximum likelihood estimates $\{u(x,y), v(x,y)\}$ by minimizing the SSD

$$E = \sum_{x,y} [I_1(x + u(x,y), y + v(x,y)) - I_0(x,y)]^2 \qquad (7)$$

between the reference image $I_0(x,y)$, and the predicted image $I_1(x + u(x,y), y + v(x,y))$.

3.1 Wavelet Motion Model

We will approximate motion vectors $u(x,y)$ and $v(x,y)$ by using two-dimensional basis functions. A natural extension of one-dimensional to two-dimensional basis functions are using the tensor product. Accordingly, the two-dimensional basis functions

are (Figure 3)

$$\Phi_{0,k_1,k_2}(x,y) = \phi(x-k_1)\phi(y-k_2) \quad (8)$$
$$\Psi^H_{j,k_1,k_2}(x,y) = \phi(2^j x-k_1)\psi(2^j y-k_2) \quad (9)$$
$$\Psi^V_{j,k_1,k_2}(x,y) = \psi(2^j x-k_1)\phi(2^j y-k_2) \quad (10)$$
$$\Psi^D_{j,k_1,k_2}(x,y) = \psi(2^j x-k_1)\psi(2^j y-k_2) \quad (11)$$

where the subscripts j, k_1 and k_2 represent the resolution scale, horizontal and vertical translations, respectively, and the upper subscript H, V and D represent the horizontal, vertical and diagonal directions, respectively. Similar to the one-dimensional case, any two dimensional motion vector can be expressed in terms of linear combinations of coarsest-scale spline function (8) and horizontal, vertical and diagonal wavelets ((9), (10) and (11)) in finer levels[8]

$$u(x,y) = u_{-1}(x,y) + \sum_{j=0}^{J}(u_j^H(x,y) + u_j^V(x,y) + u_j^D(x,y))$$
$$v(x,y) = v_{-1}(x,y) + \sum_{j=0}^{J}(v_j^H(x,y) + v_j^V(x,y) + v_j^D(x,y))$$

where

$$u_{-1}(x,y) = \sum_{k_1=-2}^{L_1-2}\sum_{k_2=-2}^{L_2-2} c_{-1,k_1,k_2}\Phi_{0,k_1,k_2}(x,y)$$
$$u_j^H(x,y) = \sum_{k_1=-2}^{2^jL_1-2}\sum_{k_2=-1}^{2^jL_2-2} c^H_{j,k_1,k_2}\Psi^H_{j,k_1,k_2}(x,y)$$
$$u_j^V(x,y) = \sum_{k_1=-1}^{2^jL_1-2}\sum_{k_2=-2}^{2^jL_2-2} c^V_{j,k_1,k_2}\Psi^V_{j,k_1,k_2}(x,y)$$
$$u_j^D(x,y) = \sum_{k_1=-1}^{2^jL_1-2}\sum_{k_2=-1}^{2^jL_2-2} c^D_{j,k_1,k_2}\Psi^D_{j,k_1,k_2}(x,y).$$

Functions v_{-1}, v_j^H, v_j^V and v_j^D have similar forms except that c_{-1,k_1,k_2}, c^H_{j,k_1,k_2}, c^V_{j,k_1,k_2} and c^D_{j,k_1,k_2} are replaced by d_{-1,k_1,k_2}, d^H_{j,k_1,k_2}, d^V_{j,k_1,k_2} and d^D_{j,k_1,k_2}, respectively.

3.2 The Algorithm

Given I_0 and I_1, the optical flow estimation is now a problem of estimating coefficients vector $\hat{\mathbf{c}} = [(\hat{\mathbf{c}}_{-1})^T \cdots \ (\hat{\mathbf{c}}_j^{H,i})^T \ \ (\hat{\mathbf{c}}_j^{V,i})^T \ \ (\hat{\mathbf{c}}_j^{D,i})^T \cdots]^T$ where $\hat{\mathbf{c}}_{-1}$ representing the coarsest-scale coefficients, and $\hat{\mathbf{c}}_j^{H,i}$, $\hat{\mathbf{c}}_j^{V,i}$ and $\hat{\mathbf{c}}_j^{D,i}$ representing any finer-scale coefficients in horizontal, vertical and diagonal directions, respectively. This is done by substituting $u(x,y)$ and $v(x,y)$ into (7) and minimizing SSD with respect to $\hat{\mathbf{c}}$ iteratively. The motion vectors are reconstructed using the estimated coefficients. We then use the motion vectors to warp image I_1 toward I_0. In the next finer scale, we estimate the coefficients from coarsest to current scale where the large-to-small regions are utilized. To handle large displacement, we first estimate the coarsest-scale coefficients where the largest regions are used.

Let vector $\hat{\mathbf{c}}^i_{-1} = [c^i_{-1,-2,-2}, \cdots, c^i_{-1,L_1-2,L_2-2}, d^i_{-1,-2,-2}, \cdots, d^i_{-1,L_1-2,L_2-2}]^T$ be the current estimated coarsest-scale coefficients during the ith iteration. The incremental estimate $\delta\hat{\mathbf{c}}_{-1}$ can be obtained by minimizing a quadratic measure using the differential method. By using the first order Taylor series expansion

$$I(x+u^i+\delta u, y+v^i+\delta v) \approx I(x+u^i, y+v^i, t) + \delta u I_x + \delta v I_y,$$

the incremental quadratic error is

$$E(\delta\hat{\mathbf{c}}_{-1}) = E(\hat{\mathbf{c}}^i_{-1}+\delta\hat{\mathbf{c}}_{-1}) - E(\hat{\mathbf{c}}^i) \approx \delta\hat{\mathbf{c}}^T_{-1} A \delta\hat{\mathbf{c}}_{-1} - 2\mathbf{b}^T\delta\hat{\mathbf{c}}_{-1} \quad (12)$$

where

$$(I_x, I_y) \stackrel{\text{def}}{=} \left(\frac{\partial I_1(x+u^i, y+v^i)}{\partial x}, \frac{\partial I_1(x+u^i, y+v^i)}{\partial y}\right)$$
$$A \stackrel{\text{def}}{=} \sum_{x,y}\mathbf{a}(x,y)\mathbf{a}(x,y)^T, \quad \text{Hessian matrix} \quad (13)$$
$$\mathbf{a}(x,y) \stackrel{\text{def}}{=} [\Phi_{0,-2,-2}I_x \cdots \Phi_{0,L_1-2,L_2-2}I_x \vdots \Phi_{0,-2,-2}I_y \cdots \Phi_{0,L_1-2,L_2-2}I_y]^T, \quad (14)$$
$$\mathbf{b} \stackrel{\text{def}}{=} -\sum_{x,y}(I_1(x+u^i, y+v^i) - I_0(x,y))\mathbf{a}(x,y),$$
$$u^i(x,y) = u^i_{-1}(x,y), \ v^i(x,y) = v^i_{-1}(x,y) \quad (15)$$

To obtain Hessian matrix A and gradient vector $\mathbf{b}$ during each iteration in every resolution scale, we first calculate the warped image $I_1(x+u^i, y+v^i)$ using the updated motion vector $u^i(x,y)$ and $v^i(x,y)$, and bilinear interpolation. The gradient (I_x, I_y) and the residual difference, $I_1(x+u^i, y+v^i) - I_0(x,y)$, are then computed. When $E(\delta\hat{\mathbf{c}}_{-1})$ is minimized using the Levenberg-Marquardt algorithm[6], the resultant coefficients $\hat{\mathbf{c}}^i_{-1}$ are used as the initial guess and propagated into the next finer level.

In the next finer level, we update the coefficients estimated at the previous coarsest level, and estimate three sets of coefficients at the current level one by one. First, we denote the first set of coefficients at current level ($j=0$) by $\hat{\mathbf{c}}_0^{H,i} = [c^{H,i}_{0,-1,-1} \cdots c^{H,i}_{0,L_1-2,L_2-2} \vdots d^{H,i}_{0,-1,-1} \cdots d^{H,i}_{0,L_1-2,L_2-2}]^T$, and estimate $\hat{\mathbf{c}}^i = [(\hat{\mathbf{c}}^i_{-1})^T \ (\hat{\mathbf{c}}_0^{H,i})^T]^T$ where $\hat{\mathbf{c}}^i_{-1}$ has been initialized from the previous step. The coefficients $\hat{\mathbf{c}}^i$, which have more local influence, are used to reconstruct the motion vectors $u(x,y) = u_{-1}(x,y) + u_0(x,y)$ and $v(x,y) = v_{-1}(x,y) + v_0(x,y)$. The incremental estimate $\delta\hat{\mathbf{c}}$ can be obtained by minimizing the same form of quadratic cost function (12) except that new terms $\Psi^H_{0,-2,-1}I_x \cdots \Psi^H_{0,L_1-2,L_2-2}I_x$, $\Psi^H_{0,-2,-1}I_y \cdots \Psi^H_{0,L_1-2,L_2-2}I_y$ are included in $\mathbf{a}(x,y)$, and $(u_0^{H,i}(x,y), v_0^{H,i}(x,y))$ are include in $(u^i(x,y), v^i(x,y))$. More precisely, they are

$$\mathbf{a}(x,y) \stackrel{\text{def}}{=} [\Phi_{0,-2,-2}I_x \cdots \Phi_{0,L_1-2,L_2-2}I_x \vdots$$

$$\Psi^H_{0,-2,-1} I_x \cdots \Psi^H_{0,L_1-2,L_2-2} I_x \vdots$$
$$\Phi_{0,-2,-2} I_y \cdots \Phi_{0,L_1-2,L_2-2} I_y \vdots$$
$$\Psi^H_{0,-2,-1} I_y \cdots \Psi^H_{0,L_1-2,L_2-2} I_y]^T$$

$$u^i(x,y) = u^i_{-1}(x,y) + u^{H,i}_0(x,y), v^i(x,y) = v^i_{-1}(x,y) + v^{H,i}_0(x,y)$$

When $E(\delta\hat{\mathbf{c}})$ attains minimum, the resultant coefficients $\hat{\mathbf{c}}$ are propogated into the next step where the second set coefficients are estimated.

Then, we estimate the second set of coefficients $\hat{\mathbf{c}}^{V,i}_0 \stackrel{\text{def}}{=} [c^{V,i}_{0,-1,-2} \cdots c^{V,i}_{0,L_1-2,L_2-2} \vdots\ d^{V,i}_{0,-1,-2} \cdots d^{V,i}_{0,L_1-2,L_2-2}]^T$, and update the previous estimated coefficients $\hat{\mathbf{c}}^i_{-1}$ and $\hat{\mathbf{c}}^{H,i}_0$ simultaneously. The concatenated coefficient vector is $\hat{\mathbf{c}}^i = [(\hat{\mathbf{c}}^i_{-1})^T\ (\hat{\mathbf{c}}^{H,i}_0)^T\ (\hat{\mathbf{c}}^{V,i}_0)^T]^T$. The incremental estimate $\delta\hat{\mathbf{c}}$ can be obtained by minimizing the same form of quadratic cost function (12), except that new terms $\Psi^V_{0,-1,-2} I_x \cdots \Psi^V_{0,L_1-2,L_2-2} I_x$ and $\Psi^V_{0,-1,-2} I_y \cdots \Psi^V_{0,L_1-2,L_2-2} I_y$ are added to previous $\mathbf{a}(x,y)$, and $(u^{V,i}_0(x,y), v^{V,i}_0(x,y))$ are added to previous $(u^i(x,y), v^i(x,y))$. When $E(\delta\hat{\mathbf{c}})$ attains minimum, the resultant coefficients $\hat{\mathbf{c}}$ are propagated into the next step where the third set coefficients are estimated.

Lastly, we estimate the third set of coefficients $\hat{\mathbf{c}}^{D,i}_0 \stackrel{\text{def}}{=} [c^{D,i}_{0,-1,-1} \cdots c^{D,i}_{0,L_1-2,L_2-2} \vdots\ d^{D,i}_{0,-1,-1} \cdots d^{D,i}_{0,L_1-2,L_2-2}]^T$, and update the previous estimated coefficients $\hat{\mathbf{c}}^i_{-1}$, $\hat{\mathbf{c}}^{H,i}_0$ and $\hat{\mathbf{c}}^{V,i}_0$. The concatenated coefficient vector is $\hat{\mathbf{c}}^i = [(\hat{\mathbf{c}}^i_{-1})^T\ (\hat{\mathbf{c}}^{H,i}_0)^T\ (\hat{\mathbf{c}}^{V,i}_0)^T\ (\hat{\mathbf{c}}^{D,i}_0)^T]^T$. The incremental estimate $\delta\hat{\mathbf{c}}$ can be obtained by minimizing the same form of quadratic cost function except, that new terms $\Psi^D_{0,-1,-1} I_x \cdots \Psi^D_{0,L_1-2,L_2-2} I_x$ and $\Psi^D_{0,-1,-1} I_y \cdots \Psi^D_{0,L_1-2,L_2-2} I_y$ are added to previous $\mathbf{a}(x,y)$, while $(u^{D,i}_0(x,y), v^{D,i}_0(x,y))$ are added to previous $(u^i(x,y), v^i(x,y))$.

In general, at level j, we estimate the coefficients from the coarsest to the current level. At each level $(j \geq 0)$, three sets of coefficients are estimated one by one. The estimation procedure is the same as we described above. When $E(\delta\hat{\mathbf{c}})$ attains minimum, we check the SSD prediction error and decide to either stop or continue the process.

3.3 Variably Sized Supports and Hessian Matrix

The structure of Hessian matrix reveals the usage of global and local information and how they are interactive with each other. The structure of Hessian matrices using one, two and three levels are shown in Figure 4. The nonzeros and zeros elements are presented as white and black regions, respectively. We see that each Hessian matrix can be divided into four patterns, say A_{11}, A_{12}, A_{21} and A_{22} representing left upper, right upper, left lower and right lower patterns, respectively. Let us denote the first and second half parts of vector $\mathbf{a}$ by $\mathbf{a_1}$ and $\mathbf{a_2}$ which are associated with coefficients c_j's and d_j's, respectively. Patterns A_{11} and A_{22} are formed by $\sum_{x,y} \mathbf{a_1}\mathbf{a_1}^T$ and $\sum_{x,y} \mathbf{a_2}\mathbf{a_2}^T$, respectively, pattern A_{12} is formed by $\sum_{x,y} \mathbf{a_1}\mathbf{a_2}^T$, and A_{12} is equal to A_{21}. A_{11} and A_{12} are used to calculate coefficients c_{j,k_1,k_2}'s, and A_{21} and A_{22} are used to calculate coefficients d_{j,k_1,k_2}'s. Each entry in the Hessian matrix is produced by the multiplications of any two basis functions with each other, and with two image gradients. As a result, the entry is nonzero when the associated two basis functions are overlapped and two image gradients are nonzeros. In Figure 4, we observe that as the resolution scale moves up, the basis functions have narrower supports and fewer overlaps with functions in coarser and current scale. Therefore, the white regions becomes sparser. Since the block diagonal nonzero entries of any pattern are produced by any two overlapped basis functions at the same scale and the remaining blocks are produced by the overlaps of different basis functions from coarsest to current scales. This implies that in the finer scales not only smaller but also larger patches are optimally used in the sense that the coarser- and finer-scale coefficients minimize SSD.

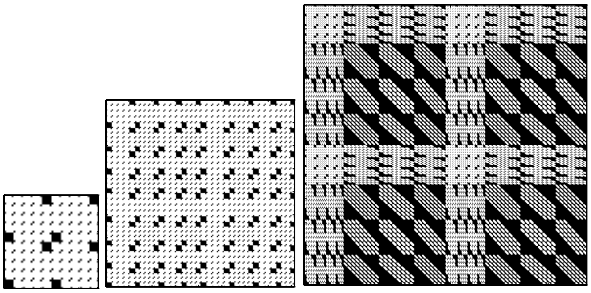

Figure 4: The left, middle and right figures are the forms of hessian matrix when one, two and three levels are used. White and black regions represent nonzero and zero entries, respectively.

4 Experimental Results

This section compares the results produced by wavelet-based and spline-based methods.

4.1 Synthetic Image Pairs Containing Small Displacement

In the first experiment, we use the synthetic images in paper [1], where the ground truth of optical flow vectors is provided. In the pyramid spline-based method, the low-pass Gaussian filter with mask size 3 $\times$ 3 pixels is successively applied to create the coarse-to-fine blurred images. Two error measurements we use are the angular measure θ_e [1] and the magnitude measure m_e, defined by

$$\theta_e = \arccos\left(\frac{uu_t + vv_t + 1}{\sqrt{u^2+v^2+1}\sqrt{u_t^2+v_t^2+1}}\right) \quad (16)$$

$$m_e = \left|\sqrt{u^2+v^2} - \sqrt{u_t^2+v_t^2}\right| \quad (17)$$

Image Pairs	Spline-based method		Wavelet-based method	
	$\overline{\Delta\theta_e}$	$\overline{\Delta m_c}$	$\overline{\Delta\theta_c}$	$\overline{\Delta m_c}$
Sinusoid 1	0.22^0 (1)*	0.54 (1)	0.40^0 (1)	0.03 (1)
	71.37^0 (3)	19.043 (3)	0.056^0 (3)	0.0021 (3)
Translating	13.27^0 (1)	0.79 (1)	0.45^0 (1)	0.026 (1)
Tree	0.59^0 (3)	0.0210 (3)	0.85^0 (3)	0.030 (3)
Diverging	4.23^0 (1)	0.065 (1)	1.33^0 (1)	0.0254 (1)
Tree	1.52^0 (3)	0.0254 (3)	2.49^0 (3)	0.030 (3)
Yosemite	4.38^0 (3)	0.136 (3)	4.63^0 (3)	0.137 (3)
			3.54^0 (4)	0.097 (4)

*The number inside the parentheses denotes the image level(s) used in the spline-based method, or the motion resolution level(s) in wavelet-based method.

where $(u_t(x,y), v_t(x,y))$ is the correct motion vector. For each image pair, we compute the averaged angle and magnitude errors ($\overline{\Delta\theta_e}$ and $\overline{\Delta m_c}$), using the flow vector at each pixel. We have tested the image pairs of **Sinusoidal 1**, **Translating Tree**, **Diverging Tree** and **Yosemite**. Figures 5-8 show the estimated motion vectors using our method. To compare our method with pyramid spline-based methods, the number of unknowns is designed to be the same. We name the flow estimates produced by spline-based and by our methods as spline and wavelet flow, respectively. The quantitative comparisons between pyramid splin-based and wavelet-based methods are shown in the Table. In the **Sinusoid 1** image pair, the one level spline flow is more accurate than the one level wavelet flow but less accurate than the three level wavelet flow. The three level spline flow has large errors, $(71.37^0, 19.043)$, which are propagated from the coarsest estimates due to the significant texture change. In the **Translating Tree** and **Diverging Tree** image pairs, one level spline flow is less accurate than the wavelet flow because of incorrect estimates around the poor texture regions. In the **Yosemite** image pair, the three level spline flow is better than the wavelet flow. However, we obtained better results when 4 levels were used. In general, both flow estimates are accurate and comparable, and both methods outperform other methods in [1] and method in [7]. (see [4] and [8]).

4.2 Synthetic Image Pair Containing Large Displacement

In the second experiment, the input image (Figure 9) is created by cropping part of the **Sinusoidal 1** and the **Tree** images where the sinusoidal pattern has a displacement of 1.585 pixels upward and 0.863 pixel rightward, and the tree pattern has 10 pixels displacements to the right. The results in the left column of Figure 10 show that large errors occurring at level 1 due to spatial aliasing are propagated to the final result. The results using wavelet method are shown in the right column of Figures 10. The flow vectors of the **sinusoidal 1** and **Tree** patterns are well recovered.

4.3 Real Image Sequences

The results of three real image sequences, SRI, NASA, Hamburg Taxi sequences in [1] are presented in Figures 11, 12, and 13, respectively. Flow of ten image frames is presented for each image sequence. Overall, the results look reasonable. Finally, the left and right columns in Figure 14 show the computed optical flow representing surprised and smile expressions with head movement using the wavelet method. The optical flow results of 6 basic facial expressions can be seen in *www.cs.cmu.edu/afs/cs.cmu.edu/user/ytw/www/facial.html* and [8].

5 Conclusions

We have used the wavelet model to develop a multiple-window, coarse-to-fine matching algorithm. The wavelet basis functions play triple roles in our algorithm, allowing us to construct motion vectors from coarse-to-fine, select multiple large-to-small regions for image matching and impose smoothness. Since the motion vector at each pixel is obtained by simultaneously matching large-to-small full resolution regions, our algorithm produces robust and accurate results. The advantage has been demonstrated by comparing the results obtained by our method with those obtained by Szeliski pyramid spline method. This method has been applied successfully to the motion estimation of facial expressions[8], and we are in the process of extending our algorithm to other applications.

Acknowledgements

This research is supported by NIMH grant R01MH51435. We would also like to thank Marie Elm for reviewing the manuscript.

References

[1] Barron, J.L., Fleet, D.J., and Beauchemin, S.S. *Performance of optical flow techniques*, International Journal of Computer Vision, vol. 12, 1994, pp. 43-77.

[2] Bergen, J. R., Anandan, P., Hanna, K. J., and Hingorani R. *Hierarchical model-based motion estimation*, in: G. Sandini, ed., Computer Vision-ECCV '92,Springer, Berlin, 1992, pp. 237-252.

[3] Horn, B.K.P. and Schunck, B.G. *Determining optical flow*: A retrospective, Artificial Intelligence, vol. 17, 1981, pp. 185-203.

[4] Szeliski, R. and Coughlan, J. *Spline-Based Image Registration* International Journal of Computer Vision, 22(3), 1997, pp. 199-218.

[5] Cai, W. and Wang J. *Adaptive multiresolution collocation methods for initial boundary value problems of nonlinear PDEs*, SIAM NUMER. ANAL. Vol. 33, No. 3, pp. 937-970, June 1996.

[6] Press, W. H., Flannery, B. P., Teukolsky, S. A., and Vetterling,W. T. *Numerical Recipes in C: The Art of Scientific Computing*, Cambridge University Press, Cambridge, 2nd, 1992.

[7] Xiong, Y. and Shafer, S. A. *Moment and Hypergeometric Filters for High Precision Computation of Focus, Stereo and Optical Flow*, International Journal of Computer Vision, vol. 22(1), 1997, 25-59.

[8] Yu-Te Wu, Image Registration Using Wavelet-Based Motion Model and Its applications, PhD Dissertation, University of Pittsburgh, 1997.

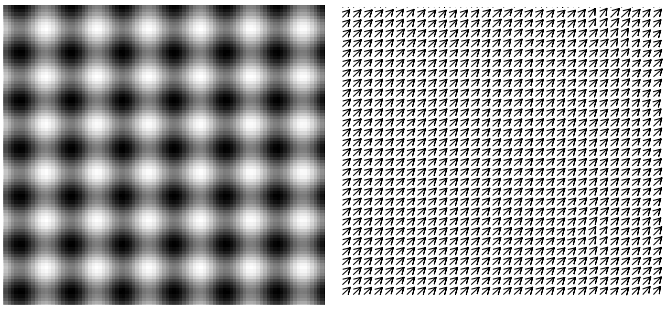

Figure 5: Sinusoidal 1

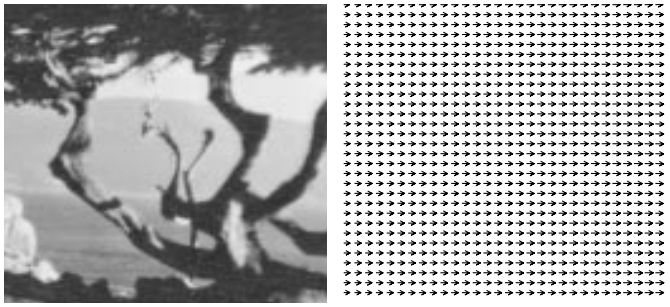

Figure 6: Translating Tree

Figure 7: Diverging Tree

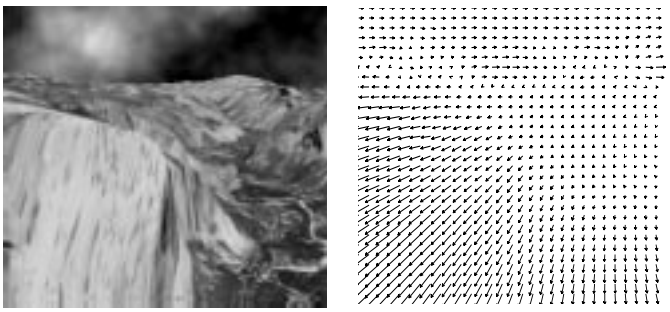

Figure 8: Yosemite

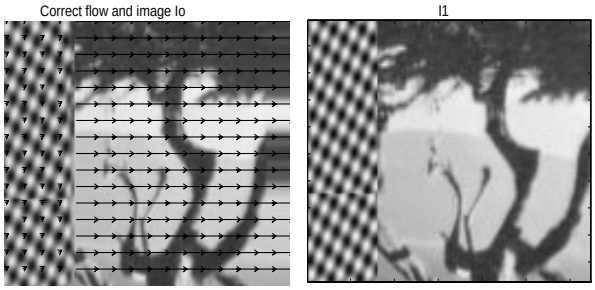

Figure 9: Snythetic image pair (128× 128) containing large and small motions. The sinusoidal pattern (left) has displacement 1.585 pixel upward and 0.863 pixel rightward, and the tree pattern (right) has 10 pixels displacements to the right.

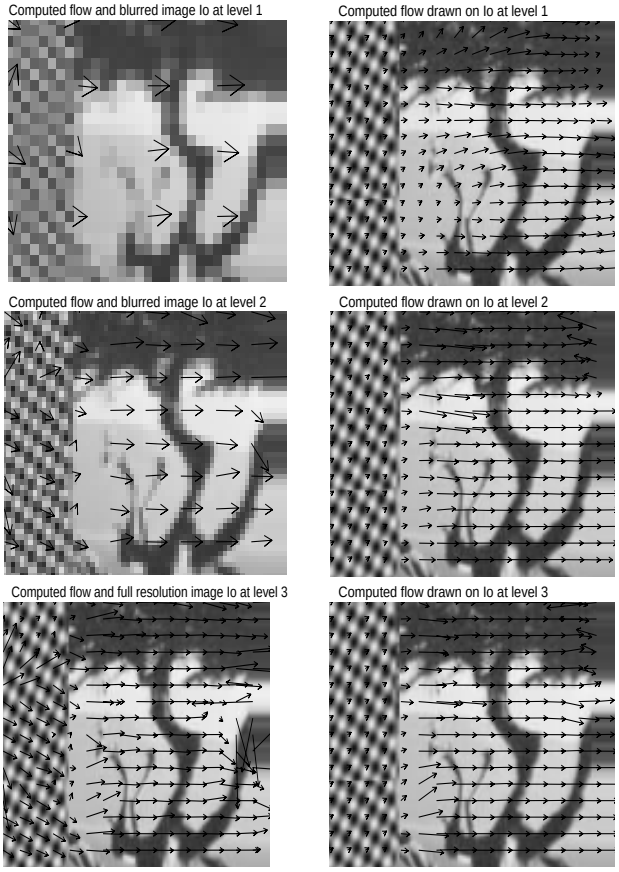

Figure 10: The comparison of pyramid spline-based (left column) and wavelet-based method (right column). Observe that in left column the large errors which occurred in the left **Sinusoidal 1** textured region at level 1 have been wrongly propagated to finer estimates. In the right column, the optical flow are recovered correctly..

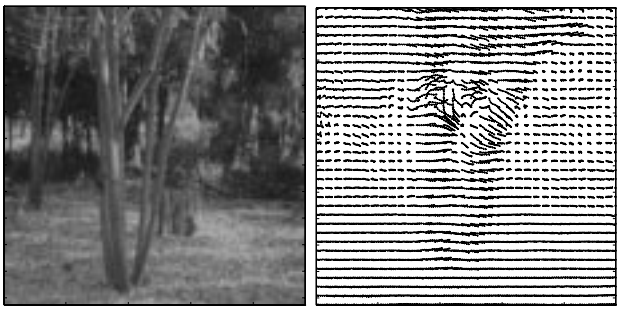

Figure 11: SRI Trees.

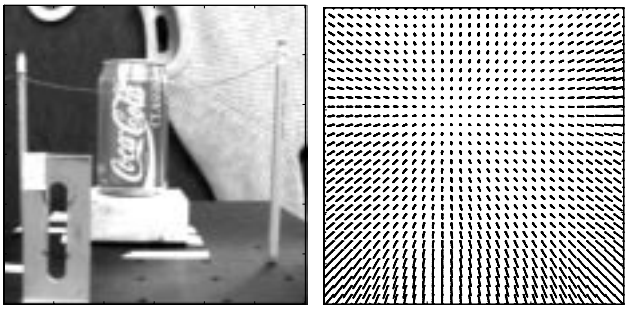

Figure 12: NASA

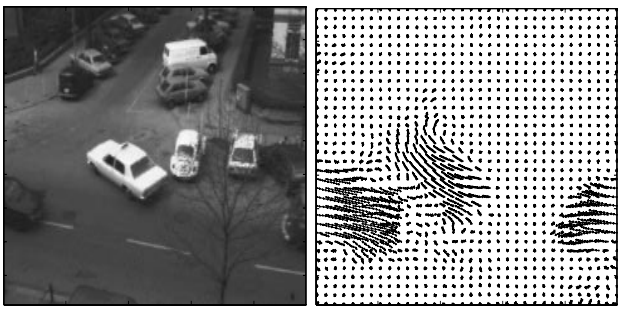

Figure 13: Hamburg Taxi.

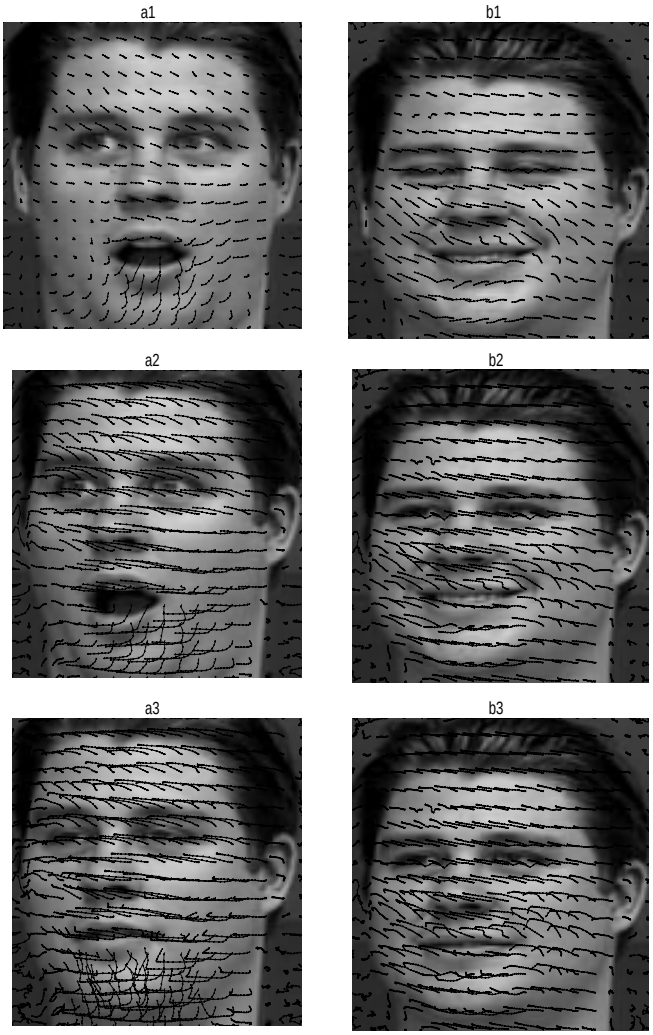

Figure 14: Two image sequences (left and right columns) containing large displacements. Each sequence has 25 frames. Three selected frames are shown from each sequence.