Eng. & Technology, Vol.25, Suppl. of No.2, 2007

3D Wavelet-Based Optical Flow Estimation

Dr. Mutaz. S. Abdul Wahab

Received on: 30/8/2005
Accepted on: 25/1/2006

Abstract

In this paper, a new algorithm for accurate optical flow estimation using discrete wavelet approximation is proposed. The image sequences are always assumed to be noiseless in the computation of optical flow, since there is always a method that can perform such task. One of the main application areas of the wavelet transform is that of noise reduction in images. The basic technique is to transform the noisy input image into a domain, in which the main signal energy is concentrated into as few coefficients as possible, while the noise energy is distributed more uniformly over all coefficients. The choice of the transform is represent an important tool in optical flow estimation. In this paper, several algorithms of 1-D, 2-D and 3-D wavelet transforms are adapted for the estimation of optical flow for the first time.

Keyword Optical flow estimation, gradient-based method, 3-D Discrete Wavelet Transform (DWT).

استخلاص التدفق البصري باستخدام تحويل المويجة المتقطعة للاشارة ثلاثية الابعاد

الخلاصة

يُقتَرح في هذ البحث طريقة جديدة لاستخلاص التدفق البصريّ باستخدام تحويل المويجة. يفترض على الدوام ان الصور المتتابعة تكون خالية من الضوضاء عند حساب التدفق البصري.حيث توجد طريقة للقيام بذلك. واحد من اهم تطبيقات تحويل المويجة هو خفض مستوى الضوضاء في الصور. مبدا هذه التقنية هو تحويل الادخال الصاخب الى مجال بحيث ان معظم طاقة الاشارة تتركز في اقل عدد ممكن من المعاملات، بينما الطاقة التي تحتوي على ضوضاء تتوزع بشكل منتظم اكثر على جميع المعاملات. ان اختيار نوع التحويل مهم جدا عند حساب التدفق البصري. يُقدِّمُ في هذ البحث بعض الخواروميات لحساب تحويل المويجات بالنسبة للاشارات ذات البعد الواحد والبعدين والثلاثة ابعاد. بالاضافة الى ذلك، ولاول مرة، يتم اقتراح طريقة جديدة في حساب التدفق البصري بالاعتماد على تحويل المويجات للاشارات ثلاثية الابعاد.

* Dept. of Electrical and Electronic Eng., UOT., Baghdad, IRAQ.

Introduction

Optical flow (OF) estimation is an essential problem in motion analysis of image sequences. It provides information needed for video technology, such as object tracking, image segmentation, and motion compensation. A great number of approaches for OF estimation has been proposed in the literature, including gradient-based, region-based, energy-based, and wavelet-based techniques [1-4]. Practical issues in computing optical flow were addressed in [5], in which nine techniques are reported dating from 1981 to 1990 for accuracy, density and reliability of measurements. One of the purposes of [5] was to analyze the performance of different optical flow methods and to encourage others to compare numerical results with theirs. In addition, some experimental work evaluating differential techniques has recently appeared [6].

Despite their difference, many of these techniques can be viewed conceptually in terms of three stages of processing [5]:

- Prefiltering or smoothing with low pass or band pass filters in order to extract signal structures of interest and to enhance the signal-to-noise ratio.
- Measurement extraction of the basic image structures, such as spatiotemporal derivatives or local correlation surfaces.
- The integration of these measurements either by regularization, correlation, or a least-squares computation to produce a 2D flow field, which often involves assumptions about the smoothness of the underlying flow field.

A typical gradient-based approach was proposed by Horn and Schunck [1], which is mainly based on optimizing an energy function that is a function of an image constraint and a smoothness constraint

$$E = \iint \left[\left(I_x u + I_y v + I_t \right)^2 + l \left(\left| \nabla u \right|^2 + \left| \nabla v \right|^2 \right) \right] dxdy \quad \ldots (1)$$

where

$I = I(x,y,t)$ image brightness function at time t.

$[u,v] = [u(x,y),v(x,y)]$ the flow vector;

∇ = gradient operator; $\lambda >= 0$ is the regularizing parameters.

I_x , I_y , I_t = partial derivatives of I(x,y,t) with respect to the x and y coordinates, and time , respectively.

The first term on the right-hand side of equation (1) is the image constraint, the second term is the smoothness constraint, that is the weighting between the two constraints. For each pixel (x,y), two variables, u and v, need to be solved. With only one constraint (the image constraint), the solution of u and v cannot be obtained. Thus, Horn and Schunck proposed the smoothness constraint and added it into the objective function shown in equation (1). Under these circumstances, the flow field can be solved by optimizing the objective function. It is known that the smoothness constraint may be invalid across the motion boundary, but this problem can be solved by using the regularization technique [5]. Another major concern is the approximation errors that occur when the gradient-

based approach is adopted. These errors are due to inaccurate numerical approximation of partial derivatives, as well as temporal and spatial aliasing during sampling of the image brightness function *I(x,y,t)*. In order to solve the above-mentioned error problem, Barron *et al*. [5] proposed an improved approach based on equation (1). They suggested applying a spatiotemporal pre-smoothing to the target image first. Then, a four-point central difference technique is adopted to simulate the differentiation operation. Their method significantly improved the method proposed by Horn and Schunck [1] because of adding the smoothing step.

1. Discrete Wavelet Transform

The multiresolution idea is better understood by using a function represented by $\Phi(t)$ and referred to as scaling function. A two-dimensional family of functions is generated, from the basic scaling function by [7]:

$$\Phi_{j,k}(t) = 2^{j/2}\,\Phi\left(2^{j}t - k\right) \qquad \ldots (2)$$

The nesting of the space spanned by $\Phi\left(2^{j}t-k\right)$ is achieved by requiring $\Phi(t)$ to be represented by the space spanned by $\Phi(2t)$. In this case, the lower resolution function, $\Phi(t)$, can be expressed by a weighted sum of shifted version of the same scaling function at the next higher resolution, $\Phi(2t)$, as follows:

$$\Phi(t) = \sum_{k} h(k)\,\sqrt{2}\,\Phi(2t-k) \qquad \ldots (3)$$

The set of coefficients $h(k)$ being the scaling function coefficients and $\sqrt{2}$ maintains the norm of the scaling function with scale of two. $\Phi(t)$, being the scaling function which satisfies this equation, is sometimes called the refinement equation, the dilation equation, or the multiresolution analysis equation (MRA) [8-9].

The important features of a signal can be better described or parameterized, not by using $\Phi_{j,k}(t)$ and increasing j to increase the size of the subspace spanned by the scaling functions, but by defining a slightly different set of functions $\Psi_{j,k}(t)$ that span the differences between the spaces spanned by the various scales of the scaling function [10]. Since it is assumed that these wavelets reside in the space spanned by the next narrower scaling function, they can be represented by a weighted sum of shifted version of the scaling function $\Phi(2t)$ as follows:

$$\Psi(t) = \sum_{k} g(k)\,\sqrt{2}\,\Phi(2t-k) \qquad \ldots (4)$$

The set of coefficients $g(k)$'s is called the wavelet function coefficients (or the wavelet filter). It is shown that the wavelet coefficients are required by orthogonality to be related to the scaling function coefficients for a finite even length-N, by [8,10]:

$$g(k) = (-1)^{k}\,h(N-1-k) \qquad \ldots (5)$$

Any function $f(t)$ could be written as a series expansion in terms

of the scaling function and wavelets by [11]:

$$f(t)=\sum_{k=-\infty}^{\infty} a_{j_0}(k)\Phi_{j_0,k}(t)+\sum_{j=j_0}^{\infty}\sum_{k=-\infty}^{\infty} b_j(k)\Psi_{j,k}(t) \quad \dots (6)$$

where N : Length of filter.
j_o : coarse scale.
a_j : Scaling coefficients.
b_j : Wavelet coefficients.

It is shown that the scaling and wavelet coefficients at scale j are related to the scaling coefficients at scale $(j + 1)$ by the two following relations.

$$a_j(k)=\sum_m h(m-2k)\ a_{j+1}(m) \dots (7)$$

$$b_j(k)=\sum_m g(m-2k)\ a_{j+1}(m) \dots (8)$$

2. A Proposed Fast Computation Method of DWT

For computing fast discrete wavelet transform (FDWT), consider the following transformation matrix for length-2 [12]:

$$T=\begin{bmatrix} h(0) & h(1) & 0 & 0 & \mathbf{L} & \mathbf{L} & & \\ 0 & 0 & h(0) & h(1) & \mathbf{L} & \mathbf{L} & & \\ \mathbf{M} & \mathbf{M} & \mathbf{M} & \mathbf{M} & \mathbf{L} & \mathbf{L} & & \\ 0 & 0 & 0 & 0 & \mathbf{L} & \mathbf{L} & h(0) & h(1) \\ g(0) & g(1) & 0 & 0 & \mathbf{L} & \mathbf{L} & & \\ 0 & 0 & g(0) & g(1) & \mathbf{L} & \mathbf{L} & \mathbf{M} & \mathbf{M} \\ \mathbf{M} & \mathbf{M} & \mathbf{M} & \mathbf{M} & \mathbf{L} & \mathbf{L} & & \\ 0 & 0 & 0 & 0 & \mathbf{L} & \mathbf{L} & g(0) & g(1) \end{bmatrix} \quad \dots (9)$$

and the following transformation matrix for length-4:

$$T=\begin{bmatrix} h(0) & h(1) & h(2) & h(3) & 0 & 0 & \mathbf{L} & 0 & 0 & 0 & 0 \\ 0 & 0 & h(0) & h(1) & h(2) & h(3) & \mathbf{L} & 0 & 0 & 0 & 0 \\ \mathbf{M} & \mathbf{M} & \mathbf{M} & \mathbf{M} & \mathbf{M} & \mathbf{M} & \mathbf{L} & \mathbf{M} & \mathbf{M} & \mathbf{M} & \mathbf{M} \\ h(2) & h(3) & 0 & 0 & 0 & 0 & \mathbf{L} & 0 & 0 & h(0) & h(1) \\ g(0) & g(1) & g(2) & g(3) & \mathbf{M} & \mathbf{M} & \mathbf{L} & 0 & 0 & 0 & 0 \\ 0 & 0 & g(0) & g(1) & g(2) & g(3) & \mathbf{L} & 0 & 0 & 0 & 0 \\ \mathbf{M} & \mathbf{M} & \mathbf{M} & \mathbf{M} & \mathbf{M} & \mathbf{M} & \mathbf{L} & \mathbf{M} & \mathbf{M} & \mathbf{M} & \mathbf{M} \\ 0 & 0 & 0 & 0 & 0 & 0 & \mathbf{L} & g(0) & g(1) & g(2) & g(3) \\ g(2) & g(3) & 0 & 0 & 0 & 0 & \mathbf{L} & 0 & 0 & g(0) & g(1) \end{bmatrix} \quad \dots (10)$$

Here blank entries signify zeros. By examining the transform matrices of the scalar wavelet as shown in equations (9) and (10) respectively. One can see that, the first row generates one component of the data convolved with the low-pass filter coefficients ($h(0)$, $h(1)$, …), likewise the second, third, and other upper half rows. The lower half rows perform a different convolution, with high pass filter coefficients ($g(0)$, $g(1)$, …). Thus overall action of the matrix, is to perform two related convolutions, then to decimate each of them by half (throw away half the values), and interleave the remaining halves.

By using equation (5), the transform matrices become:

$$T=\begin{bmatrix} h(0) & h(1) & 0 & 0 & \mathbf{L} & \mathbf{L} & & \\ 0 & 0 & h(0) & h(1) & \mathbf{L} & \mathbf{L} & & \\ \mathbf{M} & \mathbf{M} & \mathbf{M} & \mathbf{M} & \mathbf{L} & \mathbf{L} & & \\ 0 & 0 & 0 & 0 & \mathbf{L} & \mathbf{L} & h(0) & h(1) \\ h(1) & -h(0) & 0 & 0 & \mathbf{L} & \mathbf{L} & & \\ 0 & 0 & h(1) & -h(0) & \mathbf{L} & \mathbf{L} & \mathbf{M} & \mathbf{M} \\ \mathbf{M} & \mathbf{M} & \mathbf{M} & \mathbf{M} & \mathbf{L} & \mathbf{L} & & \\ 0 & 0 & 0 & 0 & \mathbf{L} & \mathbf{L} & h(1) & -h(0) \end{bmatrix} \quad \dots (11)$$

$$T=\begin{bmatrix} h(0) & h(1) & h(2) & h(3) & 0 & 0 & \mathbf{L} & 0 & 0 & 0 & 0 \\ 0 & 0 & h(0) & h(1) & h(2) & h(3) & \mathbf{L} & 0 & 0 & 0 & 0 \\ \mathbf{M} & \mathbf{M} & \mathbf{M} & \mathbf{M} & \mathbf{M} & \mathbf{M} & \mathbf{L} & \mathbf{M} & \mathbf{M} & \mathbf{M} & \mathbf{M} \\ h(2) & h(3) & 0 & 0 & 0 & 0 & \mathbf{L} & 0 & 0 & h(0) & h(1) \\ h(3) & -h(2) & h(1) & -h(0) & \mathbf{M} & \mathbf{M} & \mathbf{L} & 0 & 0 & 0 & 0 \\ 0 & 0 & h(3) & -h(2) & h(1) & -h(0) & \mathbf{L} & 0 & 0 & 0 & 0 \\ \mathbf{M} & \mathbf{M} & \mathbf{M} & \mathbf{M} & \mathbf{M} & \mathbf{M} & \mathbf{L} & \mathbf{M} & \mathbf{M} & \mathbf{M} & \mathbf{M} \\ 0 & 0 & 0 & 0 & 0 & 0 & \mathbf{L} & h(3) & -h(2) & h(1) & -h(0) \\ h(1) & -h(0) & 0 & 0 & 0 & 0 & \mathbf{L} & 0 & 0 & h(3) & -h(2) \end{bmatrix} \quad \dots (12)$$

For such characterization to be useful, it must be possible to reconstruct the original data vector of length N from its N/2 smooth and its N/2 detail, that is effected by requiring the matrices to be orthogonal, so that its inverse is just the transposed matrix:

$$T2=\begin{bmatrix} h(0) & 0 & \mathbf{L} & 0 & h(1) & 0 & \mathbf{L} & 0 \\ h(1) & 0 & \mathbf{L} & 0 & -h(0) & 0 & \mathbf{L} & 0 \\ 0 & h(0) & \mathbf{L} & 0 & 0 & h(1) & \mathbf{L} & 0 \\ 0 & h(1) & \mathbf{L} & 0 & 0 & -h(0) & \mathbf{L} & 0 \\ 0 & 0 & \mathbf{L} & 0 & \mathbf{L} & \mathbf{L} & \mathbf{L} & 0 \\ \mathbf{M} & \mathbf{M} & \mathbf{M} & \mathbf{M} & \mathbf{L} & \mathbf{L} & \mathbf{M} & \mathbf{M} \\ 0 & 0 & 0 & h(0) & 0 & 0 & 0 & h(1) \\ 0 & 0 & 0 & h(1) & 0 & 0 & 0 & -h(0) \end{bmatrix} \quad \dots (13)$$

$$T2=\begin{bmatrix} h(0) & 0 & 0 & \mathbf{L} & \mathbf{L} & h(2) & h(3) & 0 & 0 & \mathbf{L} & 0 & h(1) \\ h(1) & 0 & 0 & \mathbf{L} & \mathbf{L} & h(3) & -h(2) & 0 & 0 & \mathbf{L} & 0 & -h(0) \\ h(2) & h(0) & 0 & \mathbf{L} & \mathbf{L} & 0 & h(1) & h(3) & 0 & \mathbf{L} & 0 & 0 \\ h(3) & h(1) & 0 & \mathbf{L} & \mathbf{L} & 0 & -h(0) & -h(2) & 0 & \mathbf{L} & 0 & 0 \\ 0 & h(2) & h(0) & \mathbf{L} & \mathbf{L} & 0 & 0 & h(1) & h(3) & \mathbf{L} & 0 & 0 \\ 0 & h(3) & h(1) & \mathbf{L} & \mathbf{L} & 0 & 0 & -h(0) & -h(2) & \mathbf{L} & 0 & 0 \\ 0 & 0 & h(2) & \mathbf{L} & \mathbf{L} & 0 & 0 & 0 & h(1) & \mathbf{L} & 0 & 0 \\ 0 & 0 & h(3) & \mathbf{L} & \mathbf{L} & 0 & 0 & 0 & -h(0) & \mathbf{L} & 0 & 0 \\ \mathbf{M} & \mathbf{M} & \mathbf{M} & \mathbf{M} & \mathbf{M} & \mathbf{M} & \mathbf{M} & \mathbf{M} & \mathbf{M} & \mathbf{M} & \mathbf{M} & \mathbf{M} \\ & & & 0 & & & & & & & 0 & 0 \\ & & & 0 & h(0) & & & & & & h(3) & 0 \\ & & & 0 & h(1) & & & & & & -h(2) & 0 \\ 0 & 0 & 0 & 0 & h(2) & h(0) & & & & & h(1) & h(3) \\ 0 & 0 & 0 & 0 & h(3) & h(1) & & & & & -h(0) & -h(2) \end{bmatrix} \quad \dots (14)$$

2.1 Computation of FDWT for 1-D Signal

To compute a single level FDWT for 1-D signal the next steps should be followed:

1. Checking input dimensions: Input vector should be of length N, where N must be power of two.
2. Construct a transformation matrix: using transformation matrices given in (11) and (12).
3. Transformation of input vector, which can be performed by applying matrix multiplication of the NxN constructed transformation matrix by the Nx1 input vector.

2.2 Computation of FDWT for 2-D Signal

To compute a single level orthogonal-based FDWT for 2-D signal the next steps should be followed:

1. Checking input dimensions: Input matrix, *X*, should be of length NxN, where N must be power of two.
2. For an NxN matrix input 2-D signal, *X*, construct a NxN transformation matrix, *T*, using transformation matrices given in eqns. (11) and (12).
3. Apply Transformation by multiplying the transformation matrix by the input matrix, and by the transpose of the transformation matrix.

$$Y = T \cdot X \cdot T^{t} \quad \ldots (15)$$

This multiplication of the three matrices results in the final discrete wavelet transformed matrix.

2.2 Computation of FDWT for 3-D Signal

The new proposed fast 3-D wavelet transform (FDWT) algorithm reduces heavily the processing time for decomposition of video sequences keeping or overcoming the quality of reconstructed sequences In addition, it cuts heavily the memory demands.

Let's take a general 3-D signal, for example any NxNxM matrix, and apply the following steps

1. Let *X* be the NxNxM input 3-D signal,

$$X=\left\{\begin{array}{cccc} x_{0,0} & x_{0,1} & x_{0,2} & x_{0,3} \\ x_{1,0} & x_{1,1} & x_{1,2} & x_{1,3} \\ x_{2,0} & x_{2,1} & x_{2,2} & x_{2,3} \\ x_{3,0} & x_{3,1} & x_{3,2} & x_{3,3} \end{array}\right.\left\{\begin{array}{cccc} y_{0,0} & y_{0,1} & y_{0,2} & y_{0,3} \\ & & & y_{1,3} \\ & & & y_{2,3} \\ & & & y_{3,3} \end{array}\right.\left\{\begin{array}{cccc} z_{0,0} & z_{0,1} & z_{0,2} & z_{0,3} \\ & & & z_{1,3} \\ & & & z_{2,3} \\ & & & z_{3,3} \end{array}\right.\left\{\begin{array}{cccc} w_{0,0} & w_{0,1} & w_{0,2} & w_{0,3} \\ & & & w_{1,3} \\ & & & w_{2,3} \\ & & & w_{3,3} \end{array}\right.$$

2. Apply 2-D FDWT algorithm to each NxN input matrix, which results in a NxNxM, Y matrix.

$$Y=\left\{\begin{array}{cccc} a_{0,0} & a_{0,1} & a_{0,2} & a_{0,3} \\ a_{1,0} & a_{1,1} & a_{1,2} & a_{1,3} \\ a_{2,0} & a_{2,1} & a_{2,2} & a_{2,3} \\ a_{3,0} & a_{3,1} & a_{3,2} & a_{3,3} \end{array}\right.\left\{\begin{array}{cccc} b_{0,0} & b_{0,1} & b_{0,2} & b_{0,3} \\ & & & b_{1,3} \\ & & & b_{2,3} \\ & & & b_{3,3} \end{array}\right.\left\{\begin{array}{cccc} c_{0,0} & c_{0,1} & c_{0,2} & c_{0,3} \\ & & & c_{1,3} \\ & & & c_{2,3} \\ & & & c_{3,3} \end{array}\right.\left\{\begin{array}{cccc} d_{0,0} & d_{0,1} & d_{0,2} & d_{0,3} \\ & & & d_{1,3} \\ & & & d_{2,3} \\ & & & d_{3,3} \end{array}\right.$$

3. Apply 1-D FDWT algorithm to each of the 16 elements in all M matrices in z-direction, which can be done as follows:

a. For each i,j , construct the Mx1 input vector

$$Y(i,j)=\left[a_{i,j} \quad b_{i,j} \quad c_{i,j} \quad d_{i,j}\right]^{T}_{1\times M}$$

where $i, j = 0,1,2,\mathbf{L},N$

b. Construct an MxM transformation matrix; using transformation matrices given in eqns. (11) and (12).

c. Apply matrix multiplication to the MxM constructed transformation matrix by the Mx1 input vector.

4. Repeat step 3 for all i, j to get YY matrix (NxNxM matrix).

3. 3D Wavelets-Based OF Estimation

Smoothing filters are used for blurring and for noise reduction. Blurring is used in preprocessing steps, such as removal of small details from an image prior to object extraction, and bridging of small gaps in lines or curves. Noise reduction can be accomplished by blurring with a linear filter and also by nonlinear filtering. Due to the sparseness of wavelet coefficients, the potential in denoising has been studied extensively over the last decade [13]. The energy of the signal is concentrated in a small number of wavelet coefficients. Thus, magnitudes of the coefficients are relatively large compared to noise which spreads over a large number of coefficients.

In this work, a new optical flow estimation approach is employed in the wavelet domain. These steps outline the computation of image velocity using 2x2x2 spatiotemporal derivative filter:

Step 1: Prefiltering: Presmoothing the images to reduce noise and aliasing effect using 3-D fast discrete wavelet transform algorithm stated in section 3.3. Only 5 input images are required, as shown in Fig. (1). When a data set is decomposed using wavelets, filters are used that act as averaging filters and other produce details. Some of the resulting wavelet coefficients correspond to details in the data set. If the details are small, they might be omitted without substantially affecting the main features of the data set.

Step 2: Estimating the partial derivatives: There are two images to estimate spatio-temporal derivatives for. A 2x2x2 spatiotemporal neighborhood, shown in Fig. (2), for estimation of partial derivatives Ix , Iy , and It is used. Each of the estimates is the average of four first differences taken over adjacent measurements in the cube:

$$I_x = \frac{1}{4}\{[I(x+1,y,t)-I(x,y,t)]+[I(x+1,y,t+1)-I(x,y,t+1)]+ \\ [I(x+1,y+1,t)-I(x,y+1,t)]+[I(x+1,y+1,t+1)-I(x,y+1,t+1)]\} \quad \ldots(16)$$

$$I_y = \frac{1}{4}\{[I(x,y+1,t)-I(x,y,t)]+[I(x+1,y+1,t)-I(x+1,y,t)]+ [I(x,y+1,t+1)-I(x,y,t+1)]+[I(x+1,y+1,t+1)-I(x+1,y,t+1)]\} \quad \ldots(17)$$

$$I_t = \frac{1}{4}\{[I(x,y,t+1)-I(x,y,t)]+[I(x+1,y,t+1)-I(x+1,y,t)]+ [I(x,y+1,t+1)-I(x,y+1,t)]+[I(x+1,y+1,t+1)-I(x+1,y+1,t)]\} \quad \ldots (18)$$

Step 3: Initialize u(x,y) and v(x,y) for all (x,y) pixel.

Step 4: Iterative Algorithm: For each iteration, the algorithm is outlined as follows:

a. Initialize ux,y and vx,y for all *(x,y)* pixel.

b. Estimate the Laplacian of the flow velocities: The Laplacian of *u* and *v* are approximated by:

$$\nabla^2 u = \bar{u}(x,y)-u(x,y)$$
$$\nabla^2 v = \bar{v}(x,y)-v(x,y) \quad \ldots (19)$$

Where $(\bar{u}^k, \bar{v}^k)$ are the neighborhood averages of (u^k, v^k)

Equivalently, the Laplacian of *u*, *v*, $\nabla^2 u$ and $\nabla^2 v$, can be obtained by applying a 3x3 window operator, shown in Fig. (3), to each point in the *u* and *v* planes, respectively.

c. Given the spatio-temporal derivatives, I_x, I_y and I_t, computed as described in step 1, a new set of velocity estimates (u^{k+1}, v^{k+1}) can be computed from the estimated derivatives and the average of the previous velocity estimates $(\bar{u}^k, \bar{v}^k)$ by:

$$u^{k+1} = \bar{u}^k - \frac{I_x[I_x\bar{u}^k + I_y\bar{v}^k + I_t]}{l^2 + I_x^2 + I_y^2}$$

$$v^{k+1} = \bar{v}^k - \frac{I_y[I_x u^k + I_y v^k + I_t]}{l^2 + I_x^2 + I_y^2} \quad \ldots (20)$$

d. Repeat until a solution is steady: there are two different ways to iterate, one is to iterate at a pixel until a solution is steady. Another way is to iterate only once for each pixel. In the latter case a good initial flow vector is required and is usually derived from the previous pixel**.**

Step 5: Thresholding: Threshold the computed velocities on the basis of the spatial intensity gradient. The typical threshold rules are to set the computed velocities smaller in absolute value than a fixed non-negative number (the threshold value, *tau*) to zero.

$$(u_{x,y}, v_{x,y}) = \begin{cases} 0.0 & if\ M(x,y) < tau \\ (u_{x,y}, v_{x,y}) & if\ M(x,y) > tau \end{cases} \quad \ldots (21)$$

where

$$M(x,y) = \|\nabla I(x,y)\| = \sqrt{I_x^2 + I_y^2} \quad \ldots (22)$$

4. Experimental Results

Algorithm developed in the previous sections are utilized and organized to be implemented in a computer package OFEM (Optical Flow Estimation Methods). The package was written in Borland C++ V.5.0, and it is developed on Pentium 4, 2.4 GHz, 128 Mbyte RAM computer. The proposed methods in this paper are applied to estimate the optical flow on real sequences and synthetic sequences. The regularizing parameter (λ) is set to 0.5 for all experiments in this work. Most image sequences are downloaded from ftp.csd.uwo.ca.

4.1 Synthetic Image Sequences

Sinusoidal Inputs: This consists of the superposition of two sinsoidal plane waves

$$\sin(k_1 \cdot x + w_1 t) + \sin(k_2 \cdot x + w_2 t) \quad \ldots (23)$$

The results reported are based on spatial wavelengths of 6 pixels, with orientations of 54º and –27º, and speeds of 1.63 and 1.02 pixel/frame respectively, which is called Sinsoidal1 as shown in Fig. (4). The resulting plaid pattern translates with velocity v=(1.5539,0.7837) pixel/frame.

Translating Squares: Other simple test case involves a translating dark square (with a width of 40 pixels) over a bright background as shown in Fig. (5).

4.2 Real Image Sequences

Two real image sequences, shown in Fig. (6) and Fig. (7), were also used:

Rotating Rubik Cube: In this image sequence a rubic's cube is rotating counterclockwise on a turntable. The motion field induced by the rotation of the cube includes velocities less than 2 pixel/frame

Hamburg Taxi Sequences: In this street scene there were four moving objects:1) the taxi turning the corner; 2) a car in the lower left, driving from left to right; 3) a van in the lower right driving right to left; and 4) a pedestrian in the upper left.

5. Conclusions

In this paper a new method for computing wavelet transforms is given. These transforms are used in several methods of optical flow estimation. On the strength of the favorable results reported by previous investigation [12-13] to apply wavelets to the denoising of 1-D and 2-D signals, combined with the success shown here for 3-D wavelets transform, it seems likely that wavelets may work well for optical flow estimation. The proposed wavelet-based optical flow estimation framework benefits from the following useful features:

a. Low computational complexity: Although the convolution method leads to full reconstruction computation, it gives also less complexity. The scalar methods have (32) multiplications and (18) additions for transforming a signal of four input data (N=4), while the

proposed method gives (16) multiplications and 12 addition operations for the same input size.

b. Low memory requirement: A Gaussian 1.5 filter requires the explicit storage of 15 frames to compute flow. Concerns arise for real-time use when considering the computational costs and seven frame delay of the Gaussian 1.5 filter. Temporal delay and storage requirements are improved significantly for wavelet filter, giving two frame latency as given in Table (1).

c. Since a significant number of motion vectors in the high frequency subbands can be zero (due to the sparse wavelets coefficients in these subbands), no motion vectors are generated for these blocks.

6. References:

[1] B. K. P. Horn and B. G. Schunck, "Determining Optical Flow", Artificial Intelligence, Vol. 17, pp.185-204, 1981.

[2] A. Singh, "An estimation-theoretic framework for image flow computation", In Proceedings of ICCV, pp. 168-177, Osaka, Japan, December, 1990.

[3] D. J. Fleet and A. D. Jepson, "Computation of component image velocity from local phase information", IJCV, Vol. 5, No.1, pp. 77-104, 1990.

[4] L. F. Chen, H. Y. M. Liao, and, J. C. Lin, "Wavelet-Based Optical Flow Estimation", IEEE Transactions on Circuits and Systems for Video Technology, Vol. 12, No. 1, January 2002.

[5] J. L. Barron, D. J. Fleet and S. S. Beauchermin, "Performance of Optical Flow Techniques", IJCV, Vol. 21, No.1, pp. 43-77, 1994.

[6] A. Bruhn, J. Weickert, C. Feddern, T. Kohlberger, and C. Schnorr, "Real-Time Optic Flow Computation with Variational Methods", CAIP 2003, LNCS 2756, pp. 222-229, 2003.

[7] A. M. Reza, "From Fourier Transform to Wavelet Transform Basic Concepts", OCTOBER 27, 1999, WHITE PAPER www.xilinxchina.com/products/logicore/dsp/fft_to_wavelet.pdf

[8] C. Valens, "A Really Friendly Guide to Wavelets", 1999, perso.wanadoo.fr/polyvalens/clemens/download/arfgtw_26022004pdf

[9] M. Suhling, M. Argovindan, P. Hunziker, and M. Unser, "Multiresolution Moment Filters: Theory and Applications", IEEE Transactions on Image Processing, Vol. 13, No. 4, April, 2004.

[10] C. S. Burrus, R. A. Gopinath, and H. Guo, "Introduction to Wavelets and Wavelet Transforms", Prentice hall, 1998.

[11] S. Rout, "Orthogonal vs. Biorthogonal Wavelets for Image Compression", M.Sc.

thesis, Virginia Polytechnic Institute and State University, Blacksburg, Virginia, august 21, 2003.

[12] N. Hadeel, "Computationally Efficient Wavelet-Based Algorithms for Optical-Flow Estimation", Ph. D. thesis, University of Technology, Baghdad, Iraq, 2005.

[13] V. Strela and A. T. Walden, "Orthogonal and Biorthogonal Multiwavelets for Signal Denoising and Image Compression", Proc. SPIE, 3391:96-107, 1998.

Table (1): Efficiency data for temporal filters.

Filter	Support (frames)	Latency (frames)
Gaussian 1.5	15	7
3D Derivative /Prefilter	7	3
2D-Wavelets	2	1
3D-Wavelets	5	2

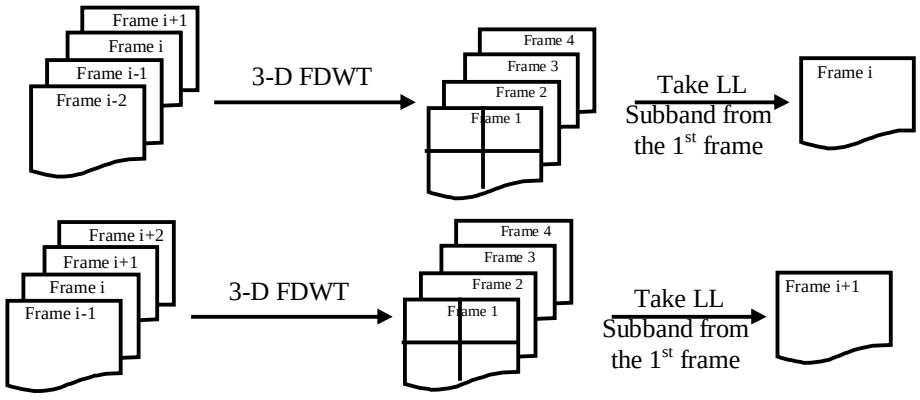

Fig. 1 Presmoothing images using 3D wavelet transform.

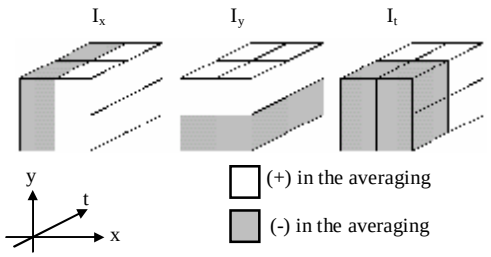

Fig. 2 Estimating *Ix* , *Iy* , and *It* using

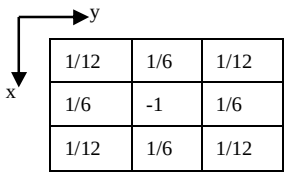

Fig. 3 A 3x3 window operation for estimation of the Laplacian of the flow vector

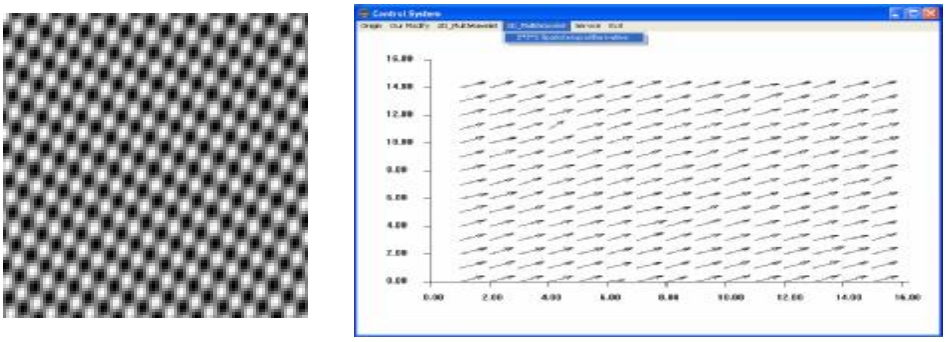

Fig. 4 Estimated optical flow obtained from the Sinusoidal image sequences

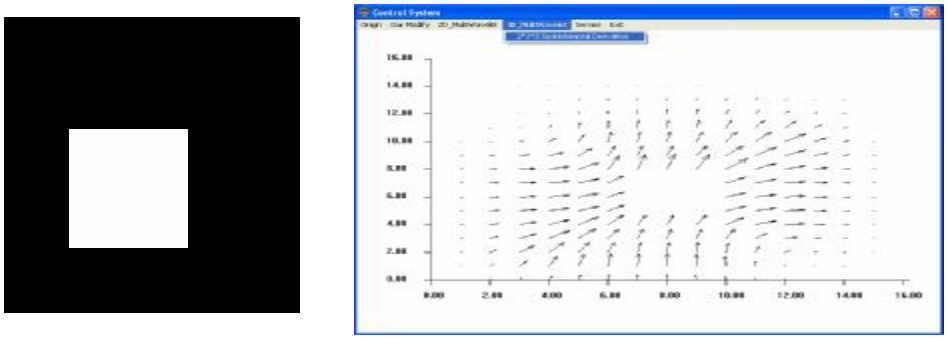

Fig. 5 Estimated optical flow obtained from the Square image sequences

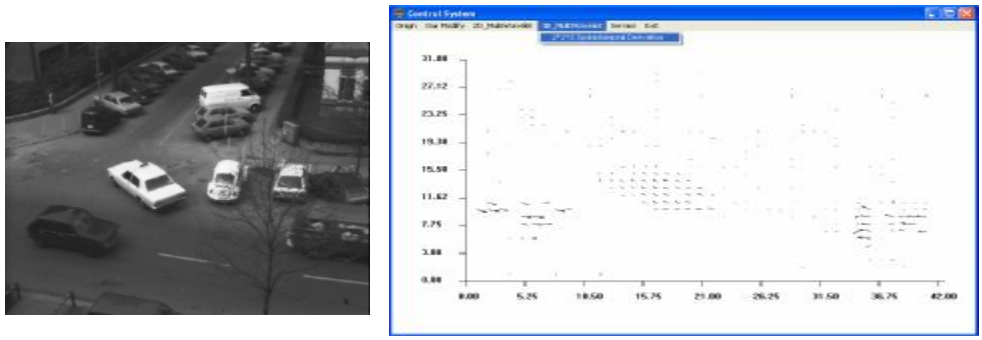

Fig. 6 Estimated optical flow obtained from the Taxi image sequences

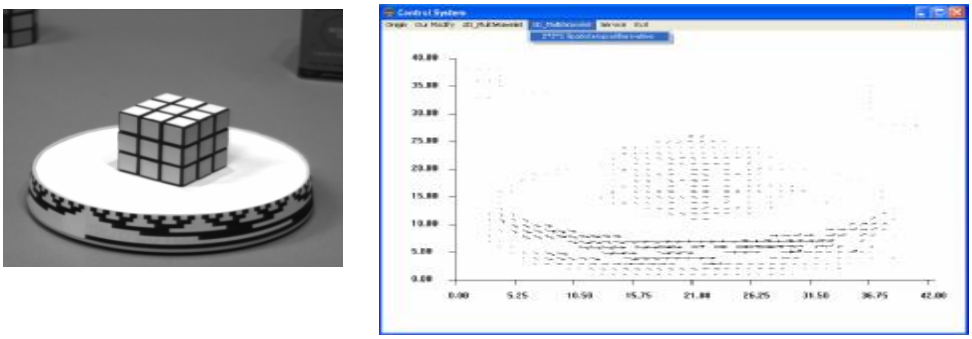

Fig. 7 Estimated optical flow obtained from the Rubic's cube image sequences