High Accuracy Optical Flow Method Based on a Theory for Warping: 3D Extension

Weixin Chen and John L. Barron

Dept. of Computer Science
The University of Western Ontario
London, Ontario, Canada, N6A 5B7
barron@csd.uwo.ca

Abstract. This paper describes the implementation and qualitative and quantitative evaluation of a 3D optical flow algorithm, whose derivation is based on the 2D optical flow method published by Brox et al. [ECCV2004]. The optical flow minimizes an energy function built with three assumptions: a brightness constancy assumption, a gradient constancy assumption, and a smoothness assumption. Brox et al. minimize the 2D version of this function using a robust estimator, which make the functional convex and guarantees convergence to a single solution. They propose a numerical solution based on nested fixed points iterations and use this scheme within a coarse-to-fine warping strategy in a 2D hierarchical image pyramid. In our 3D extension, our solution requires the regularization of a 3D function based on 3D extensions of their assumptions in a 3D hierarchical volume pyramid. We solve the corresponding Euler-Lagrange equations iteratively using nested iterations. We present 3D quantitative results on three sets of 3D sinusoidal data (with and without motion discontinuities), where the correct 3D flow is known. We also present a qualitative evaluation on the 3D flow computed using gated MRI beating heart sequence.

Keywords: 2D and 3D optical flow, regularization, brightness constancy constraint, gradient constancy constraint, Horn and Schunck smoothness constraint, hierarchical volume pyramid, 3D reverse warping, quantitative and qualitative error analysis.

1 Introduction

Starting with the pioneering work of Horn and Schunck (1981) [1] and Lucas and Kanade (1981) [2], many methods have been proposed to estimate optical flow. Brox et al. [3] presented a variational model for computing 2D optical flow that integrates three constraints: brightness constancy and global smoothness (both proposed by Horn and Schunck [1]) and gradient constancy. They employed an image warping technique and showed that warping is a good way to handle large displacements in a multi-resolution pyramid. Their published quantitative results are still the best for the Yosemite image sequence. Indeed, they obtain near perfect flow for the clouds in the Yosemite images by using heavy smoothing [4]. These clouds are fractal-based deformable objects and should not lend themselves to an accurate flow recovery. Later, Papenberg et al. [5] extended this method and investigated additional constancy constraints. These only make the flow slightly better so we ignore them in our algorithm.

This paper presents the design, implementation and analysis of a 3D extension of the 2-frame 2D Brox et al. algorithm in MatLab to see if we can obtain good 3D flow using

3D volumetric data sequences [6]. We quantitatively evaluate our algorithm on three datasets of synthetic sinusoidal data (with both continuous and discontinuous motions) and we also perform qualitative evaluation on a gated MRI cardiac dataset [7,8,6].

2 3D Optical Flow Algorithm

Brox et al.'s algorithm [3] uses several constraints which we extend to 3D. The **Grayvalue Constancy Constraint** requires the grayvalue distribution about a voxel move with that voxel. That is, $I(x,y,z,t) = I(x+u, y+v, z+w, t+1)$, where I denotes a 4D volumetric dataset and $(u,v,w,1)$ is the displacement vector between an image (when we say image from now on we mean 3D image or volume) at times t and $t+1$. Performing a 1^{st} order Taylor series expansion of $I(x,y,z,t)$, yields the **optical flow constraint** (sometimes called the **motion constraint**t) equation in 3D, $\nabla I \cdot (u,v,w) + I_t = 0$, where $\nabla I = (I_x, I_y, I_z)$ is the spatial intensity gradient at (x,y,z). The **Gradient Constancy Constraint** is a constraint that ensures I is invariant under small grayvalue changes by requiring $\nabla I(x,y,z,t) = \nabla I(x+u, y+v, z+w, t+1)$. These constraints only consider displacement of a voxel locally without taking into account the interaction between neighboring voxels. Therefore, if the gradient vanishes somewhere or if the flow can only be computed in the direction normal to the gradient (this is the aperture problem), the model presented so far will not work. As a result, we need to include a spatial Horn and Schunck like **Smoothness Constraint** to attenuate these problems and give a globally smooth flow field. This constraint requires $|\nabla u|^2 + \nabla v|^2 + |\nabla w|^2$ be as near 0 as possible. Finally, a hierarchical pyramid of the images is constructed with smoothing and downsampling using a small reduction factor, $\eta \approx 0.95$. We use MatLab's function **imresize** to build the pyramid from 2 original Gaussian blurred images $(\sigma = 1.3)$. The use of a pyramid allows faster motions to be computed using a coarse to fine strategy (motions are slower the further up the pyramid you go), where at each pyramid level, the flow between the 2 images is computed and use to reverse warp the 2^{nd} image into the 1^{st} image: if the flow is good, the 1^{st} image and the warped second image will closely agree. Warping, effectively, removes the motion from the 2^{nd} image.

2.1 The Energy Function

The grayvalue and gradient constraints are measured by the energy:

$$\begin{aligned} E_{Data}(u,v,w) = \int_{\Omega} \Psi \big(&|I(x+u, y+v, z+w, t+1) - I(x,y,z,t)|^2 \\ &+\gamma\, |\nabla I(x+u, y+v, z+w, t+1) - \nabla I(x,y,z,t)|^2 \big)\, dxdydzdt. \end{aligned} \quad (1)$$

Here Ψ does robust estimation by ensuring the function is convex (a single global solution results) and $\Psi(s^2) = \sqrt{s^2 + \epsilon}$. Brox et al. use $\epsilon = 0.001$. A smoothness term that models the assumption of piecewise smoothness is:

$$E_{smooth}(u,v,w) = \int_{\Omega} \Psi \left(|\nabla u|^2 + |\nabla v|^2 + |\nabla w|^2 \right) dxdydzdt. \quad (2)$$

Here $\nabla = (\partial_x, \partial_y, \partial_z)$ is the spatio-temporal gradient. The total energy is the weighted sum between the data term and the smoothness term:

$$E(u,v,w) = E_{Data} + \alpha E_{smooth}, \quad (3)$$

with regularization parameter $\alpha > 0$. Now the computational task is to find the values u, v and w that minimizes this function. Brox et al. derive the Euler-Lagrange equations (which are nonlinear) and then use a nested iteration scheme to minimize these. The velocity at iteration $a+1$, $(u^{a+1}, v^{a+1}, w^{a+1})$, is computed from the velocity at iteration a as (u^a, v^a, w^a) + (du^a, dv^a, dw^a), where (du^a, dv^a, dw^a) is the iteration improvement velocity vector. The idea is to initially set (du^a, dv^a, dw^a) to $(0, 0, 0)$, compute Ψ'_{Data} and Ψ'_{Smooth} using these values at the start of the outer iterations and then to iteratively refine (du^a, dv^a, dw^a) in an inner iteration to best satisfy these Ψ' values. Then these new (du^a, dv^a, dw^a) values are used to update Ψ'_{Data} and Ψ'_{Smooth} and modify (u^a, v^a, w^a) for the next step in the outer iteration, after which (du^a, dv^a, dw^a) are zero to $(0, 0, 0)$ again and then iteratively re-computed in another inner iteration using these new Ψ' values. This outer iteration and all the inner iterations terminate when either a specified maximum number of iterations is reached (typically 100 or 200) or the difference between the computed vector between two adjacent iterations is less than a threshold TOL (we use 10^{-3} in this paper).

Processing starts at the highest level in the pyramid. This computed flow field is then projected down one level in the pyramid (see details in Section 2.3) and used to inverse warp the second image into the first image (see detail in Section 2.4). The idea is that the measured motion between the 1^{st} and 2^{nd} images is "removed". Of course, the flow field used to do the warping is not precisely correct. Flow is now computed between the 1^{st} image and the inverse warped 2^{nd} image and this correction flow field is added to the projected flow field from the higher pyramid level to obtain the new flow field for that level. Projection and warping are continually performed until the final level in the pyramid (the original image) is reached. Note that this processing allows "fast" motion to be handled because at the higher levels in the pyramid the motion is slowed by the image size reduction and the warping calculation prevents the motion from increasing in magnitude as processing descends the pyramid. Full mathematical development, including all the equations and their derivations and pseudo code for the nested iterations algorithm are available in a M.Sc. thesis [6].

2.2 Intensity Differentiation

Brox et al. used 4 point central differences to compute spatial derivatives I_x, I_y and I_z using the kernel $(0.0866, -0.6666, 0.0, 0.6666, -0.0866)$. Second order derivatives are computed using the same kernel applied on the first order derivatives. Temporal derivatives are computed as simple voxel differences. In image processing nomenclature, this is called a 2 point difference and is known to be a poor approximation to a temporal derivative.

2.3 Projecting Velocities Between Adjacent Levels

Note that the size (including the width, the height and the depth) of images at different pyramid levels are not the same. Therefore, we use MatLab function `imresize` to use resample the new flow field (with rescaling by multiplication of this new flow field by expansion factor $\frac{1}{\eta}$) to go from a coarser level to the next finer level in the pyramid.

2.4 3D Inverse Warping

Given a flow field for an image $I(t)$, $\boldsymbol{v}(t) = (u(t), v(t), w(t))$ and the 2^{nd} image $I(t+1)$ in the sequence we can compute the 1^{st} image using inverse warping. That is, for floating point image location $(i + u(t), j + v(t), k + w(t))$, one can use tri-linear interpolation via MatLab function `interp3` on the grayvalues at the 8 surrounding neighbourhood integer image locations in the 2^{nd} image to compute the grayvalue at integer location (i, j, k) in the 1^{st} image. The reverse nature of this calculation ensures each pixel in the interpolated 1^{st} image gets a grayvalue (if $(i+u(t), j+v(t), k+w(t))$ is outside the image boundaries the interpolated value is set to 0).

3 Generation of 3D Sinusoid Volume Datasets

The main advantage of sinusoidal sequences is that both the flow fields and the 1^{st} and 2^{nd} order derivatives can be computed precisely, allowing quantitative analysis of the algorithm's performance. We generate three sinusoid volume datasets: **sinL**, **sinR** and **sin**. Each volume contains 31 slices of 256×256 data (unsigned shorts in the range [0-4095], i.e. 12 bits). The **sin** data is a combination of **sinL** and **sinR** which have different velocities in its left part and right part. We generate **sinL** and **sinR** using:

$$sin(\boldsymbol{k}_1 \cdot \boldsymbol{p} + \omega_1 t) + sin(\boldsymbol{k}_2 \cdot \boldsymbol{p} + \omega_2 t) + sin(\boldsymbol{k}_3 \cdot \boldsymbol{p} + \omega_3 t), \tag{4}$$

where $\boldsymbol{k}_i = (k_{ix}, k_{iy}, k_{iz})$ are the spatial frequencies, $\boldsymbol{p} = (x, y, z)$ are 3D spatial coordinates, ω_i is the temporal frequency and t is the temporal coordinate. We use $\boldsymbol{v}_L = (3, 2, 1)$ for the **sinL** sequence and $\boldsymbol{v}_R = (-3, -2, -1)$ for the **sinR** sequence. The **sin** dataset is made from the **sinL** and **sinR** datasets with a motion boundary separating the two sinusoids. Initially, the upper left corner point of the boundary at (120,120,10) and this is displaced by $\boldsymbol{v}_R$ at each frame. Simple differentiation of Equation (4) gives all 1^{st} and 2^{nd} order derivatives for the 3 sequences. Figure 1a shows the 10^{th} slice of the **sin.9** dataset while Figure 2 shows the correct flow.

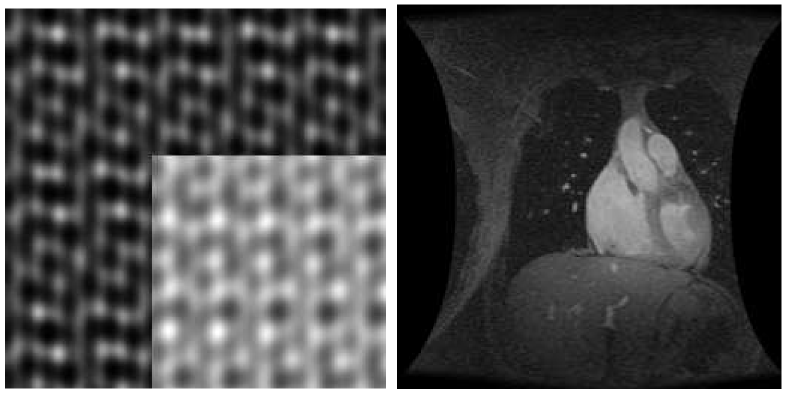

Fig. 1. (a) The 10^{th} slice of the **sin.9** sinusoid dataset and (b) the 40^{th} slice of the **10phase.9** MRI dataset.

4 3D Quantitative Error Measurement

We use the average Fleet 4D angular error measurement [9] to measure the error between the correct 3D flow field and the computed 3D flow field. Velocity can be written

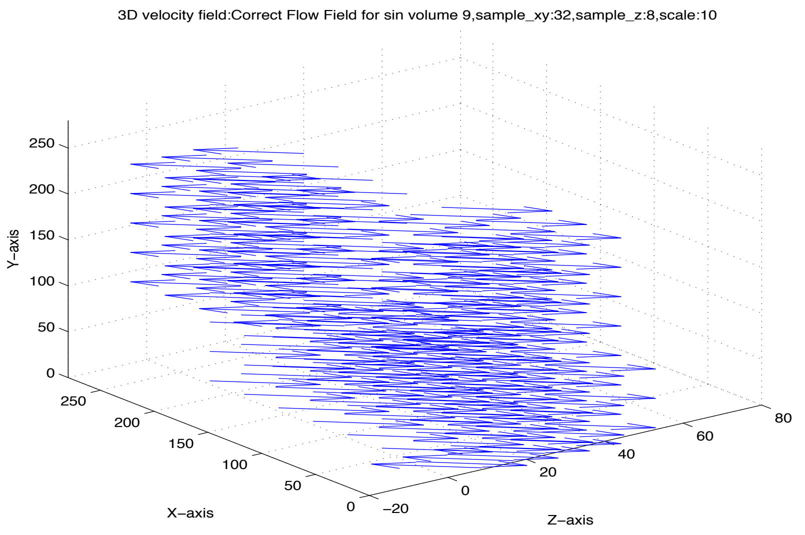

Fig. 2. The correct flow field of the 9^{th} volume of **sin** with a downsampling rate 8 in the z dimension and 32 in the x and y dimensions and with a scale factor of 10.

as displacement per time unit as in $\boldsymbol{v} = (u, v, w)$ pixels/frame or as a space-time direction vector $(u, v, w, 1)$ in units of (pixel,pixel,pixel,frame). Let $\boldsymbol{v}_c = (u_c, v_c, w_c, 1)$ represent the correct velocity at the pixel (i, j, k) and $\boldsymbol{v}_e = (u_e, v_e, w_e, 1)$ represent the computed velocity at point (i, j, k). We can compute the 4D angular error as:

$$\psi_E = \arccos\left(\frac{(u_e, v_e, w_e, 1)}{\sqrt{u_e^2 + v_e^2 + w_e^2 + 1}} \cdot \frac{(u_c, v_c, w_c, 1)}{\sqrt{u_c^2 + v_c^2 + w_c^2 + 1}}\right). \tag{5}$$

Angular error has the advantages of encoding both magnitude and direction error as 1 number and preventing zero division problems

5 Gated MRI Cardiac Datasets

Nowadays, it is possible to acquire good gated MRI (Magnetic Resonance Imagery) data of a human beating heart. The 3D motions of the heart can provide useful information for physician to detect heart disease. However, as we shall see, the measurement of 3D velocities of a beating heart is still challenging [8].

The gated MRI data we use was generated by the Robarts Research Institute at the University of Western Ontario [7]. Each set of this data contains 20 volumes of 3D data for one EGK synchronized heart beat. In our experiments, we examined the flow for one dataset, **10phase.9**, which has size $256 \times 256 \times 75$ and voxel intensities in the range $[0 - 4095]$. The MRI datasets is considerably more complex than our synthetic sinusoid datasets. A beating human heart is deformable, and its motion is discontinuous in space and time: different chambers in the heart are contracting/expanding at different times and the heart as a whole undergoes a twisting motion as it beats. The heart is also “textureless” in places, meaning good intensity derivatives are difficult to compute. The word “gated” refers to the way the data is collected: 1 or a few slices of each volume set are acquired at the same instance in a cardiac cycle. A patient lies in an MRI machine and holds his breath for approximately 42 second intervals to acquire each set of slices. This data acquisition method relies on the patient not moving or breathing during the the acquisition (this minimizes heart motion caused by a moving diaphragm or expanding/contracting lungs) [8]. Figure 1 shows the 40^{th} slice of the **10phase.9** dataset.

6 Experimental Results

Table 1 show the comparison between a basic version of 3D Horn and Schunck [8] and a version that uses Brox et al.'s hierarchical pyramid and our implementation of 3D Brox et al.'s algorithm. AAE is the abbreviation for Average Angular Error and STD is the abbreviation for Standard Deviation. Poor temporal differentiation (simple differences) is the main reason for the poor results for 3D Horn and Schunck. Error results computed using the correct derivatives are much more accurate than those computed using Brox et al.'s differentiation. For all three datasets, hierarchical 3D Horn and Schunck gives much better results than the original 3D Horn and Schunck, showing one of the benefit of using a pyramid. 3D Brox et al.'s results ($\alpha = 100, \gamma = 100$ and 10 levels of pyramid) surpasses them both. When the gradient parameter γ is turned off even better results can be obtained (see below). These results show that the worst results are for the **sin** dataset, which has significant error at the motion boundary. We sometimes use medium filtering in an attempt to remove outliers.

sinL	AAE	STD
3D Horn and Schunck (Brox derivatives, $\alpha = 100.0$)	22.89°	9.08°
3D Horn and Schunck (Correct derivatives, $\alpha = 100.0$)	0.24°	0.25°
3D Horn and Schunck ($\alpha = 100.0$,10-levels pyramid)	10.09°	6.52°
3D Brox et al.($\alpha = 100.0$,$\gamma = 100.0$,10-levels pyramid)	2.06°	2.56°
3D Brox et al.($\alpha = 100.0$,$\gamma = 0.0$,10-levels pyramid)	0.65°	1.10°
sinR	AAE	STD
3D Horn and Schunck ($\alpha = 100.0$)	23.26°	12.56°
3D Horn and Schunck (Correct derivatives, $\alpha = 100.0$)	0.78°	0.41°
3D Horn and Schunck ($\alpha = 100.0$,10-levels pyramid)	9.90°	6.87°
3D Brox et al.($\alpha = 100.0$,$\gamma = 100.0$,10-levels pyramid)	1.99°	3.55°
3D Brox et al.($\alpha = 100.0$,$\gamma = 0.0$,10-levels pyramid)	0.75°	1.56°
sin	AAE	STD
3D Horn and Schunck ($\alpha = 100.0$)	32.32°	24.49°
3D Horn and Schunck (Correct derivatives, $\alpha = 100.0$)	12.78°	24.49°
3D Horn and Schunck ($\alpha = 100.0$,10-levels pyramid)	22.67°	29.00°
3D Brox et al.($\alpha = 100.0$,$\gamma = 100.0$,10-levels pyramid)	16.20°	29.96°
3D Brox et al.($\alpha = 100.0$,$\gamma = 0.0$,10-levels pyramid)	13.11°	28.71°

Table 1. The 3D angular errors and standard deviations for flow fields with 100% density computed by 3D Horn and Schunck ($\alpha = 100.0$, 100 iterations), hierarchical 3D Horn and Schunck ($\alpha = 100.0$, 10 levels of pyramid, 100 iterations), and 3D Brox et al.'s algorithm ($\alpha = 100.0$,$\gamma = 100.0$ and $\gamma = 0.0$, 10 levels in the pyramid, outer iterations=1, inner iterations=100, all with $3 \times 3 \times 3$ median prefiltering, for **sinL.9**, **sinR.9** and **sin.9**

6.1 Inner and Outer Iterations and Convergence

Another experiment we performed with the correct derivatives was to see the effect of the inner and outer iteration on the result. The outer iterations update Ψ'_{data} and Ψ'_{smooth}, which are used in Equations (1) and (2) [6]. The inner iterations update the increments in velocities du, dv and dw. We can define the motion constraint term $eqID$, and gradient constraint terms, $eqIx$, $eqIy$ and $eqIz$, as:

$$eqID = (I_D + I_x du + I_y dv + I_z dw)^2, \quad (6)$$
$$eqIx = (I_{xD} + I_{xx} du + I_{xy} dv + I_{xz} dw)^2, \quad (7)$$
$$eqIy = (I_{yD} + I_{yx} du + I_{yy} dv + I_{yz} dw)^2, \quad (8)$$
$$eqIz = (I_{zD} + I_{zx} du + I_{zy} dv + I_{zz} dw)^2. \quad (9)$$

Then, the arguments to Ψ'_{Data} can be written as $\Psi'\left(eqID + \gamma(eqIx + eqIy + eqIz)\right)$. As velocities get better, Equation (6) and Equations (7), (8) and (9) should become close to 0. Tables 1 shows the average change in Ψ'_{data} and Ψ'_{smooth}, the average values of $eqID$, $eqIx$, $eqIy$ and $eqIz$, the average angle error and standard deviation for 10 inner iterations for the 1^{st}, the 5^{th} and the 10^{th} outer iteration of **sin** using Brox et al.'s spatial derivatives and temporal differences. We can see that as we perform more iterations, $eqID$, $eqIx$, $eqIy$ and $eqIz$ become smaller and smaller, which means the constraints are better satisfied. Brox et al. [3] used 10 inner and 10 outer iterations only for all their results. However, they never say why they used these number of iterations or report any investigation into the effect of this number of iterations on the flow accuracy.

The results for **sinL** and **sinR** converge quickly. By the 10^{th} outer iteration, values of these equations are almost zeros, and the average angle error goes down to $0.65°$-$0.75°$, which is very close to the correct velocities (within roundoff error). Instead, we concentrate on the **sin** data, which is our worst case synthetic dataset.

Table 2 show the average change in Ψ'_{data} and Ψ'_{smooth}, the average values of $eqID$, $eqIx$, $eqIy$ and $eqIz$, the average angle error and standard deviation for each of the 10 inner iterations for the 1^{st}, the 5^{th} and the 10^{th} outer iterations for the **sin** data using correct derivatives. [Using correct derivatives eliminates differentiation as a cause of poor performance.] In all cases, $eqID$, $eqIx$, $eqIy$ and $eqIz$ still decrease as more iteration are performed, although it is harder for them to approach 0 as there is now significant flow error at the motion boundary.

We used 1 outer iteration and 100 inner iterations for most of the results in this paper, because this gives much better results than using different sets of numbers of inner and outer iterations in our implementation. Again, we emphasize that Brox et al. [3] do not investigate this behaviour and do not (necessarily) iterate until convergence. One possible reason we don't get good results when using more outer iterations may be the poor quality of the temporal derivatives used. These derivatives are used to update Ψ'_{data} and Ψ'_{smooth} in each outer iteration which, in turn, may propagate errors in further iterations.

6.2 Hierarchical Optical Flow

One way to understand the Brox et al. method is to see how fields change at different pyramid levels. Figures 3 and 4 show the start and end flow fields for **sin.9** for levels

1^{st} Outer Iteration							
Inner	Ψ'_{data} diff.	$\Psi'_{sm.}$ diff.	ave_eqID	ave_eqIx	ave_eqIy	ave_eqIz	AEE.±STD.
1	0.0005877	15.8113	170839.255	3722.804	2099.715	2897.699	$70.69° \pm 1.73°$
2	0.0000193	2.0166	159501.420	3455.617	1924.327	2623.482	$66.85° \pm 3.05°$
3	0.0000188	1.6025	148917.033	3230.226	1776.601	2400.909	$63.41° \pm 4.17°$
4	0.0000190	1.0764	139030.474	3031.516	1647.400	2211.502	$60.34° \pm 5.16°$
5	0.0000193	0.7289	129889.763	2853.311	1532.762	2047.496	$57.63° \pm 6.04°$
6	0.0000196	0.5160	121482.966	2691.516	1429.927	1903.577	$55.23° \pm 6.83°$
7	0.0000199	0.3808	113774.957	2543.410	1336.984	1776.070	$53.12° \pm 7.56°$
8	0.0000202	0.2914	106717.469	2407.037	1252.514	1662.261	$51.25° \pm 8.24°$
9	0.0000205	0.2304	100257.859	2280.929	1175.425	1560.079	$49.61° \pm 8.86°$
10	0.0000208	0.1866	94343.514	2163.938	1104.849	1467.893	$48.15° \pm 9.44°$
5^{th} Outer Iteration							
Inner	Ψ'_{data} diff.	$\Psi'_{sm.}$ diff.	ave_eqID	ave_eqIx	ave_eqIy	ave_eqIz	AEE.±STD.
1	0.0131558	5.8824	2724.246	20.078	28.104	34.088	$19.30° \pm 20.05°$
2	0.0347743	0.2602	2234.735	14.483	22.206	25.176	$18.60° \pm 20.12°$
3	0.0059093	0.2068	2041.729	12.709	20.032	22.482	$18.00° \pm 20.19°$
4	0.0051870	0.2228	1905.180	11.597	18.541	20.735	$17.49° \pm 20.25°$
5	0.0045679	0.2327	1804.209	10.821	17.427	19.489	$17.03° \pm 20.30°$
6	0.0043418	0.2334	1726.878	10.246	16.556	18.546	$16.62° \pm 20.34°$
7	0.0042219	0.2287	1666.133	9.807	15.859	17.809	$16.25° \pm 20.37°$
8	0.0041733	0.2211	1617.340	9.465	15.294	17.218	$15.91° \pm 20.40°$
9	0.0041689	0.2119	1577.376	9.195	14.830	16.736	$15.61° \pm 20.43°$
10	0.0041954	0.2017	1544.075	8.977	14.445	16.335	$15.32° \pm 20.45°$
10^{th} Outer Iteration							
Inner	Ψ'_{data} diff.	$\Psi'_{sm.}$ diff.	ave_eqID	ave_eqIx	ave_eqIy	ave_eqIz	AEE.±STD.
1	5.2872770	11.2821	684.577	5.147	7.791	9.594	$6.90° \pm 18.75°$
2	0.6563276	0.0441	667.738	5.040	7.599	9.421	$6.80° \pm 18.71°$
3	0.3178679	0.0422	658.751	5.011	7.546	9.373	$6.71° \pm 18.67°$
4	0.2265391	0.0396	651.884	4.991	7.507	9.332	$6.64° \pm 18.64°$
5	0.1905575	0.0362	646.484	4.975	7.475	9.300	$6.58° \pm 18.62°$
6	0.1744226	0.0330	642.100	4.963	7.449	9.274	$6.52° \pm 18.60°$
7	0.1672531	0.0301	638.465	4.952	7.427	9.252	$6.47° \pm 18.59°$
8	0.1642882	0.0275	635.401	4.943	7.408	9.233	$6.43° \pm 18.57°$
9	0.1626395	0.0252	632.786	4.935	7.392	9.216	$6.39° \pm 18.56°$
0	0.1600340	0.0232	630.531	4.928	7.377	9.201	$6.36° \pm 18.55°$

Table 2. Average differences in Ψ'_{data} and Ψ'_{smooth} and average values of $eqID$, $eqIx$, $eqIy$, $eqIz$, average angle error and standard deviation for 10 inner iterations of the 1^{st}, 5^{th} and 10^{th} outer iterations for **sin.9** computed with Brox et al. derivatives ($\alpha = 100$, $\gamma = 100$, number of outer iteration=10, number of inner iteration=10, 1 pyramid level with no median filtering).

10 and 1. At top of the pyramid (level 10), the size of the volume is the smallest and then both the volume size and the magnitude of the flow get larger and larger as it goes down the pyramid. Note the many erroneous flow vectors in the final flow field at the motion discontinuity at level 1. Since the correct velocities are known for the lowest pyramid image, we can compute the correct velocities at any level with a certain η value by resizing and rescaling. This allows quantitative evaluation at each level of the pyramid so we can see how the accuracy of the results evolve from level to level.

Table 3 shows the angle error for the **sin.9** data at different levels. We can see that the angular error goes down to $13.56°$ at level 7, but then increase to $16.20°$ at level 0. This is most likely because we can't compute accurate velocities in the boundary area where the 2 sinusoids, **sinL** and **sinR**, meet. Warping in this area of the image will add error and these errors will accumulate as the pyramidal processing continues. When the refinement can't suppress the errors, the angular error increases.

We also exam the effect of 3D inverse warping. The spatial difference between the 1^{st} and 2^{nd} volume images should become smaller and smaller as we descend the pyramid, if the velocities we compute are becoming more and more accurate. Figures 5a to 5c shows the difference between the 15^{th} slice of the 1^{st} and 2^{nd} volume images for the **sin.9** data at (a) the 10^{th}, (b) the 9^{th} and (c) the 1^{st} pyramid level. White means no difference while black means large difference. We can observe obvious refinement between the top two levels, but for later levels, refinements between adjacent levels are very slight and so we can hardly tell if there is any changes in the image. Therefore, we just show the final difference image at the last level. We can see that at the motion boundary there is always a large difference because we can't actually compute the correct velocities here and therefore warping in this area will be wrong.

Pyramid level	AAE $\pm$ STD	Density(%)
10	$25.12° \pm 22.27°$	100.00
9	$20.12° \pm 24.19°$	100.00
8	$15.45° \pm 24.48°$	100.00
7	$13.56° \pm 25.26°$	100.00
6	$13.85° \pm 25.60°$	100.00
5	$14.95° \pm 26.86°$	100.00
4	$16.05° \pm 28.30°$	100.00
3	$16.89° \pm 29.79°$	100.00
2	$16.55° \pm 29.92°$	100.00
1	$16.20° \pm 29.96°$	100.00

Table 3. The 3D angular errors, standard deviation and density for the flow fields at all pyramid levels of the 9^{th} volume of the **sin** data (10 levels of pyramid, $\alpha = 100.0$, $\gamma = 100.0$, number of outer iteration=1, number of outer iterations=1, number of inner iterations=100, size of median filter $(3 \times 3 \times 3)$.

Lastly, we briefly discuss the effects of variation of the various parameters of the algorithm for the **sin** data in Table 4. We can conclude:

- The algorithm works better with more levels of pyramid. It never produce the best result if it is run at just one level. But it is also possible that more levels will cause the accumulation of errors introduced by warping and interpolation.
- Larger α means more smoothing which in most cases produces better results. But this may not be true when there are motion discontinuities in the input images.
- In all of these results, the algorithm seems to return more accurate results when $\gamma = 0$, which means no gradient constraint is used. It is quite likely that the reason we get better result when the gradient constraint is turned off is because of the inaccuracy of the I_{xD}, I_{yD} and I_{zD} values which use simple temporal differences.

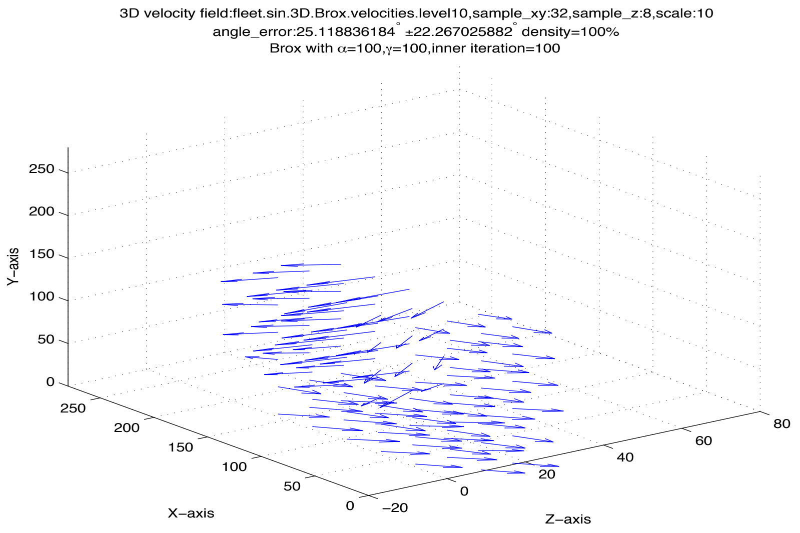

Fig. 3. The computed flow field at the 10^{th} level for **sin.9** (with $\alpha = 100.0$, $\gamma = 100.0$, number of levels in the pyramid=10, outer iteration=1, inner iteration=100, $3 \times 3 \times 3$ median filtering), downsampling rate 32 in x and y dimensions and 8 in the z dimension and flow scaling by 10.

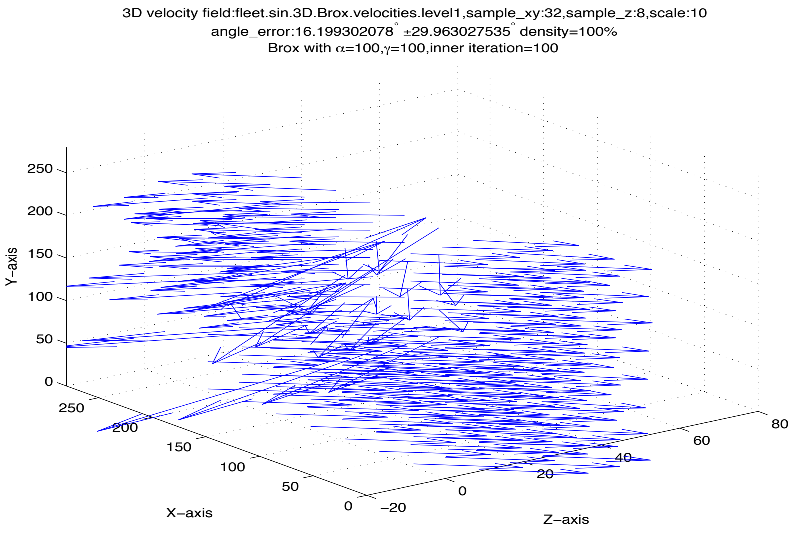

Fig. 4. The computed flow field at the 1^{st} level for **sin.9** (with $\alpha = 100.0$, $\gamma = 100.0$, number of levels in the pyramid=10, outer iteration=1, inner iteration=100, $3 \times 3 \times 3$ median filtering), downsampling rate 32 in x and y dimensions and 8 in the z dimension and flow scaling by 10.

- A median filter with larger size does remove more outliers and make the results more accurate.
- Computation costs are high. For example, with 1 outer iteration and 100 inner iterations or with 10 outer iterations and 10 inner iterations, about 2 hours of CPU

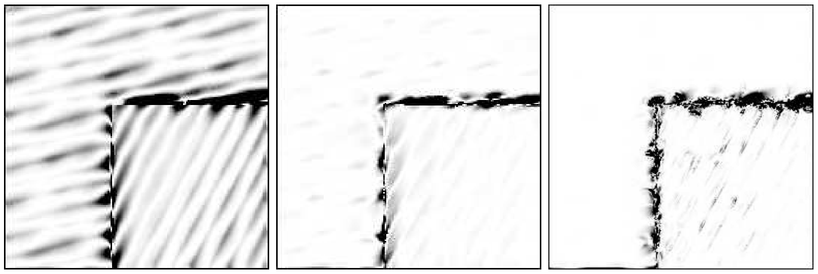
(a) (b) (c)

Fig. 5. The difference between the 15^{th} slice of the left image and the right image for the 9^{th} volume of **sin** data at (a) the 10^{th}, (b) the 9^{th}, and (c) the 1^{st} pyramid levels.

time (on a 2.8GHz PC running Linux with 8GB of RAM) was required to run our vectorized MatLab code).

Parameter				Angular Error	
Pyramid levels	α	γ	filter size	AAE.	STD.
1	100	100	3	28.50°	25.08°
10	10	100	3	30.14°	32.89°
10	100	0	3	13.11°	28.71°
10	100	20	3	14.55°	28.27°
10	100	50	3	20.89°	33.53°
10	100	100	0	17.03°	30.40°
10	100	100	3	16.20°	29.69°
30	100	0	3	12.90°	28.83°

Table 4. Error analysis for parameter variation for **sin.9**.

7 Gated MRI Cardiac Datasets Result

We note that the different parts in heart are contracting/expanding at different rates and times and the heart is undergoing a twisting as a whole. Although the heart expansion seems to have been captured, the flow field for **10phase.9** is poor, with many outliers present and is not clinically useful.

8 Conclusions and Future Work

Our 3D extension of Brox et al.'s optical flow algorithm produces very good flow for data sequences generated with continuous motion. The algorithm is not designed to handle motion discontinuities and performs poorly there. The flow field for the gated MRI cardiac dataset seems to capture some of essential motions of a beating heart but overall it is not good.

Future work includes finding out how to improve the temporal derivatives we compute and/or how to get the gradient constraint to work under poor temporal differentiation. We are investigating the parameterization of the 4D data using B-splines and a way to get good derivatives. We are currently building a functional model of the heart that will be able to predict the motion of individual parts of the heart over time. We

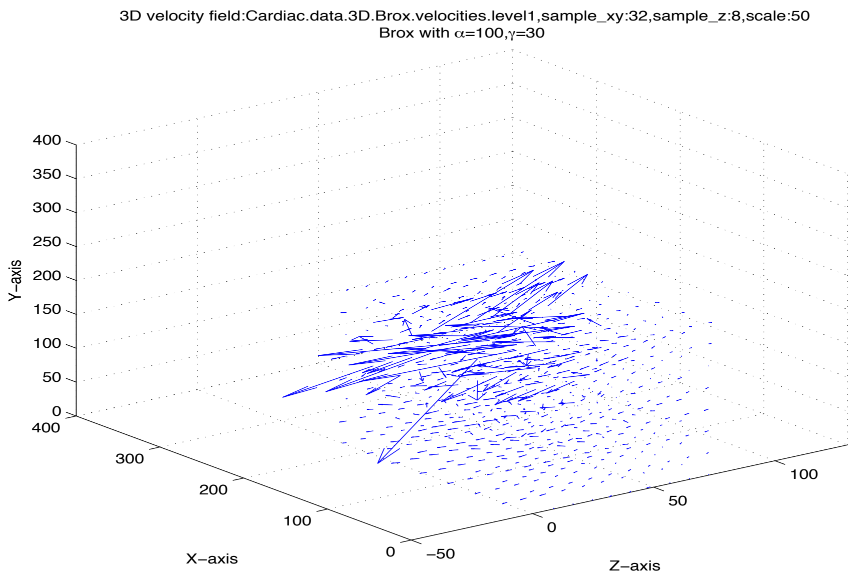

Fig. 6. The computed flow field for **10phase.9** (with $\alpha = 100$, $\gamma = 30$, number of level in the pyramid =10, outer iteration=1, inner iteration=100, $3 \times 3 \times 3$ median filtering, downsampling by 32 in the x and y dimensions and 8 in the z dimensions and a flow scale factor of 50.

would then add this as an additional constraint to our regularization. Lastly, we are investigating "Oriented Smoothness" in 3D as way to have motion discontinuities.

References

1. Horn, B., Schunck, B.: Determining optical flow. Artificial Intelligence **17**(1-3) (1981) 185–203
2. Lucas, B.D., Kanade, T.: An iterative image registration technique with an application to stereo vision. In: International Joint Conference on Artificial Intelligence. (1981) 674–679
3. Brox, T., Bruhn, A., Papenberg, N., Weickert, J.: High accuracy optical flow estimation based on a theory for warping. In: Proceedings of the 8th European Conference on Computer Vision. (2004) 25–36
4. Faisal, M., Barron, J.: High accuracy optical flow method based on a theory for warping: Implementation and qualitative/quantitative evaluation. In: Proceedings of the 4th International Conference on Image Analysis and Recognition. (2007) 513–525
5. Papenberg, N., Bruhn, A., Brox, T., Didas, S., Weickert, J.: Highly accurate optic flow computation with theoretically justified warping. International Journal of Computer Vision **67**(2) (April 2006) 141–158
6. Chen, W.: High accuracy optical flow method based on a theory for warping: 3d extension. Master's thesis, Dept. of Computer Science, The University of Western Ontario (2009)
7. Moore, J., Drangova, M., Wiergbicki, M., Barron, J., Peters, T.: A high resolution dynamic heart model. In: Medical Image Computing and Computer-Assisted Intervention (MICCAI), LNCS 2878. Volume 1. (2003) 549–555
8. Barron, J.: Experience with 3d optical flow on gated mri cardiac datasets. In: Proceedings of the 1st Canadian Conference on Computer and Robot Vision. (2004) 370–377
9. Barron, J., Fleet, D., Beauchemin, S.: Performance of optical flow techniques. International Journal of Computer Vision **12**(1) (1994) 43–77