3D Velocity from 3D Doppler Radial Velocity

J. L. Barron,[1] R. E. Mercer,[1] X. Chen,[1] P. Joe[2]

[1] Department of Computer Science, The University of Western Ontario, London, Ontario, N6A 5B7, Canada

[2] King City Radar Station, Environment Canada, Meteorological Service of Canada, Toronto, Ontario, M3H 5T4, Canada

Received 12 June 2003; accepted 31 May 2005

ABSTRACT: We present local least squares and regularization frameworks for computing 3D velocity (3D optical flow) from 3D radial velocity measured by a Doppler radar. We demonstrate the performance of our algorithms quantitatively on synthetic radial velocity data and qualitatively on real radial velocity data, obtained from the Doppler radar at Kurnell Radar station, Botany Bay, New South Wales, Australia. Radial velocity can be used to predict the future positions of storms in sequences of Doppler radar datasets.

 Int J Imaging Syst Technol, 15, 189–198, 2005; Published online in Wiley InterScience (www.interscience.wiley.com). DOI 10.1002/ima.20048

Key words: 3D optical flow; 3D Doppler full/radial velocity; local least squares; regularization

I. INTRODUCTION

We present an extension of our 2D Doppler storm tracking work (Cheng et al., 1998; Barron et al., 1999) to 3D by using local 3D radial velocity neighborhoods to compute local 3D velocity (local 3D *optical flow*). Radial velocity (measured by the Doppler effect) is the component of 3D velocity along the radial ray from the radar station to some moving 3D atmospheric point. Our computation of full velocity from atmospheric radial velocities uses a combination of a 3D local least squares framework, much like Lucas and Kanade's 2D least squares computation (Lucas and Kanade, 1981) and an iterative 3D regularization framework, much like Horn and Schunck's 2D regularization (Horn and Schunck, 1981). Preliminary results were initially presented by Chen (2001, and Chen and colleagues (2001a,b). In addition to computing 3D velocity from Doppler radial velocities, it is also possible to use densely sampled range sequences (Yamamoto et al., 1993) to compute *range flow* in a similar manner. Range flow, the 3D velocity of points on a deformable surface relative to a moving range sensor, can be computed in a similar manner (Spies et al., 2000a,b; Spies et al., 2002). Note that range flow is computed with respect to moving 3D surface data rather than with respect to moving 3D volumetric data (and is therefore not 3D optical flow) as it involves using a slightly different constraint equation (Spies et al., 2002).

2D optical flow methods have recently been generalized into the 3D domain. Chaudhury et al. (1994) formulated a 3D optical flow constraint, using I_x, I_y, I_z and I_t derivatives. Thus, they use a time-varying volume of intensity data, from which all four derivatives can be computed. Much of the 3D optical flow work has been used for medical applications, for example, to compute 3D flow for CT, MRI and PET datasets (Song and Leahy, 1991; Song et al., 1994; Zhou et al., 1995; Klein et al., 1997; Klein and Huesman, 1997). The well-known 2D motion constraint equation

$$I_x u + I_y v + I_t = 0 \tag{1}$$

forms the basis of most 2D optical flow algorithms. I_x, I_y and I_t in (1) are the x, y and t intensity derivatives while $\vec{v} = (u, v)$ is the image velocity (or optical flow) at pixel (x, y), which is an approximation of the local image motion. (1) is one equation in two unknowns and manifests the *aperture* problem. Normal velocity, the component of image velocity normal to the local intensity structure, can be totally expressed in terms of derivative information:

$$\vec{v}_{\mathrm{n}} = \frac{-I_t(I_x, I_y)^{\mathrm{T}}}{\|(I_x, I_y)\|_2^2}, \tag{2}$$

while tangential velocity, $\vec{v}_{\mathrm{t}}$, the component of image velocity tangential to the local intensity structure, cannot, in general, be recovered in a local aperture. Note that

$$\vec{v} \cdot \hat{n} = v_{\mathrm{n}} \tag{3}$$

is another equivalent way to write (1), where the direction of normal velocity is $\hat{n} = (I_x, I_y)/\|(I_x, I_y)\|_2$ and the magnitude of normal velocity is $v_{\mathrm{n}} = -I_{\mathrm{t}}/\|(I_x, I_y)\|_2$ (see Barron et al., 1994 for more details).

II. 2D OPTICAL FLOW

To resolve the aperture problem and solve $\vec{v}$, we need to impose an additional local constraint. An example of a local constraint is to assume that locally all image velocities are the same. For example,

Correspondence to: J. L. Barron; e-mail: barron@csd.uwo.ca

Grant sponsors: NSERC (National Science and Engineering Research Council of Canada) and AES (Atmospheric Environment Service).

Lucas and Kanade (1981) use a least squares computation to integrate local neighbourhoods of normal image velocities into full image velocities (here we ignore any weighting used for simplicity). For a $N = n \times n$ neighbourhood, they solve a $N \times 2$ linear system of equations $A_{N\times 2}\vec{v} = B_{N\times 1}$ as

$$\vec{v} = (A^{\mathrm{T}}A)^{-1}A^{\mathrm{T}}B, \tag{4}$$

where A has entries I_{xi} and I_{yi} in the ith row and B has entries $-I_{ti}$ in the ith row. We can perform eigenvector/eigenvalue analysis on $A^{\mathrm{T}}A$ using routines in the work of Press and colleagues (1992). Eigenvalue ($\lambda_0 \leq \lambda_1$) and corresponding eigenvector ($\hat{e}_0$ and $\hat{e}_1$) decomposition of the symmetric matrix $A^{\mathrm{T}}A$ yield least squares full image velocity, if both $\lambda_1 > \lambda_0 > \tau_{D1}$, τ_{D1} a threshold, typically 1.0 (Barron et al., 1994) or least squares normal image velocity, $\vec{v}_{\mathrm{ln}} = \vec{v} \cdot \hat{e}_1$, if $\lambda_1 > \tau_{D1}$ but $\lambda_0 \leq \tau_{D1}$ (Barron et al., 1994).

An example of a global constraint is to assume the velocity varies smoothly everywhere. For example, Horn and Schunck (Horn and Schunck, 1981) minimize

$$\iint (\nabla I \cdot \vec{v} + I_{\mathrm{t}})^2 + \lambda^2\left(\|\nabla u\|_2^2 + \|\nabla v\|_2^2\right) dx\, dy \tag{5}$$

over the entire image, where the magnitude of λ (the Lagrange multiplier) reflects the influence of the smoothness term and $\nabla I = (I_x, I_y)$ is the spatial intensity gradient. Iterative equations are used to minimize (2.5) and obtain image velocity at each image location:

$$u^{k+1} = \bar{u}^k - \frac{I_x\lfloor I_x\bar{u}^k + I_y\bar{v}^k + I_t\rfloor}{\alpha^2 + I_x^2 + I_y^2} \tag{6}$$

$$v^{k+1} = \bar{v}^k - \frac{I_y\lfloor I_x\bar{u}^k + I_y\bar{v}^k + I_t\rfloor}{\alpha^2 + I_x^2 + I_y^2}, \tag{7}$$

where k denotes the iteration number, u^0 and v^0 denote initial velocity estimates (typically zero), and $\bar{u}^k$ and $\bar{v}^k$ denote neighbourhood averages of u^k and v^k. Intensity and velocity differentiation in x, y and t can be performed using Simoncelli matched balanced lowpass and highpass filters (Simoncelli, 1994).

III. 3D OPTICAL FLOW

The 3D planar motion constraint equation can be derived in a fashion similar to the 2D motion constraint equation:

$$I_X U + I_Y V + I_Z W + I_t = 0, \tag{8}$$

where 3D velocity $\vec{V}$ has components U, V and W and I_X, I_Y, I_Z and I_{t} are the X, Y, Z and t intensity derivatives. This equation describes a plane in 3D space. It can be rewritten as

$$\vec{V} \cdot \hat{n} = V_{\mathrm{n}}, \tag{9}$$

where $\vec{V}_{\mathrm{n}} = V_{\mathrm{n}}\hat{n}$ is a 3D plane normal velocity, $\hat{n}$ is now the 3D normal direction and V_{n} is a 3D normal velocity magnitude.

$$\vec{V}_{\mathrm{n}} = \frac{-(I_X, I_Y, I_Z)I_{\mathrm{t}}}{\|(I_X, I_Y, I_Z)\|_2^2}. \tag{10}$$

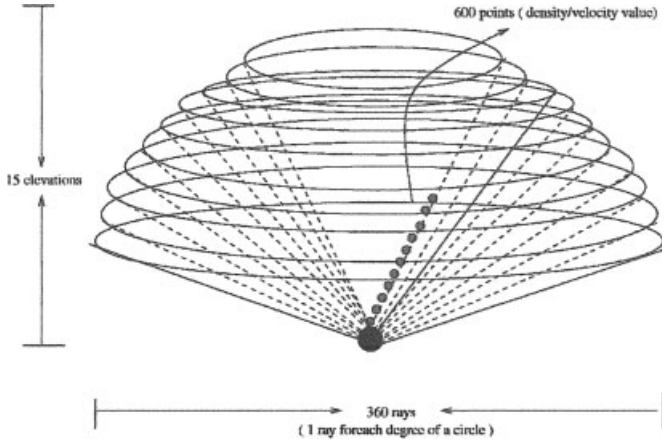


Figure 1. The 3D structure of Doppler radar datasets. The scanning surfaces of radar rays are 15 cones with different elevation angles. About 360 rays are recorded on the surface of each cone; the angle between arbitrary two adjacent rays is *one degree*. Each ray is divided into equal parts by 600 points which contains the reflectivity and the radial velocity of atmospheric water precipitation. The minimum cone angle, ϕ_{min}, is 58°, while the maximum angle, ϕ_{max}, is 89.5°. The height of the rays range from a minimum of 5.25 km and a maximum of 317.4 km, and the cone radii range from a minimum radius of 508.84 km to a maximum radius of 599.97 km. Each ray point location is described in 3D spherical polar coordinates.

The aperture problem in 3D actually yields two types of normal velocity: *plane normal* velocity, the velocity normal to a local intensity planar structure, and *line normal* velocity, the velocity normal to a local line intensity structure (Spies et al., 1999; Spies et al., 2002). In this article, we are not concerned with the computation of normal velocity, rather the Doppler radar already measures a form of normal velocity (the velocity along a radial line from the radar to some atmospheric location) directly.

IV. DOPPLER DATA

Our 3D Doppler data consist of datasets sampled at 10-min intervals. Each dataset is composed of 15 elevations of atmospheric radar reflectivity and radial velocity data. Each elevation consists of 360 equally spaced rays of such data, and each ray consists of 600 reflectivity and radial velocity data stored as unsigned characters. Figure 1 illustrates the 3D structure of Doppler radar datasets. We have implemented an X windows visualization tool to allow us to examine (and explore) the data. This program allows one not only to view the same elevation of data over time in movie mode but to view all the elevations at one time in movie mode. Also, all the elevation data at one time can be simultaneously viewed (using Z buffer hidden surface removal) and manipulated by 3D affine transformations (Chen, 2001). Our tool also performs bilinear interpolation to allow graphically pleasing views of the data. Figure 2 shows the raw and bilinearly interpolated reflectivity and radial velocity data for time 200009251510 (September 25, 15:10 hrs, 2000) at elevation two produced by our visualization program. We used the same shading (colouring) scheme as used by the Kurnell radar station to indicate radar reflectivity and radial velocity magnitude and direction in this and all other radar images in this article.

Currently these data are stored in a 3D array that consists of the 3D Cartesian coordinates of each data point, i.e., the 3D X, Y and Z

values (in kilometers) of each data point after conversion from the spherical polar coordinate format of input data.

V. 3D VELOCITY IN A LEAST SQUARES FRAMEWORK

To compute the 3D full velocity $\vec{V} = (U, V, W)$ from radial velocity $\vec{V}_r$, we use the dot product of $\vec{V}$ and the radial direction $\hat{r} = (r_X, r_Y, r_Z)$, i.e., $\vec{V} \cdot \hat{r} = V_r$, to obtain

$$Ur_X + Vr_Y + Wr_Z = V_r, \tag{11}$$

which is one equation in three unknowns. Essentially, this is the 3D motion constraint equation for 3D optical flow, i.e., (9), but with radial velocity replacing normal velocity.

To solve $\vec{V}$ at a point, we select a small local neighbourhood in the vicinity of the point. Currently, we use a $7 \times 7 \times 7$ neighbourhood (determined by trial and error). Before the least squares estimation, the data are smoothed. For each point, we calculate the average radial velocity in its $7 \times 7 \times 7$ neighbourhood, and then replace the original radial velocity at that point with its average value. Experimental results here and in a thesis (Chen, 2001) show smoothing the data before the computation and using large-sized neighbourhoods for least squares integration produce better results.

Since the neighbourhood is small compared with the whole Doppler dataset, we can assume that all points in the neighbourhood move with the same full velocity $\vec{V}$. Since the radial velocities $\vec{V}_r$ for different points satisfy different motion constraint planes, their intersection defines the common 3D full velocity $\vec{V} = (U, V, W)$, where U, V, W are three components of the velocity vector respectively along X, Y, Z axes. This forms the basis of our computation.

For a $N = n \times n \times n$ neighbourhood, we obtain a $N \times 3$ linear system of equations

$$\begin{bmatrix} r_{X_1} & r_{Y_1} & r_{Z_1} \\ r_{X_2} & r_{Y_2} & r_{Z_2} \\ \vdots & \vdots & \vdots \\ r_{X_N} & r_{Y_N} & r_{Z_N} \end{bmatrix} \begin{bmatrix} U \\ V \\ W \end{bmatrix} = \begin{bmatrix} V_{r1} \\ V_{r2} \\ \vdots \\ V_{rN} \end{bmatrix}, \tag{12}$$

which can be written as

$$A_{N\times 3}\vec{V}_{3\times 1} = B_{N\times 1},$$

where A has entries r_{X_i}, r_{Y_i} and r_{Z_i} in the ith row, B has entry V_{r_i} in the ith row and N is the number of locations in the neighbourhood. Actually, in a real computation, not all the "neighbours" have acceptable radial velocity values. We threshold out those less than a minimum radial velocity value. Only in the best case does A have the size $N \times 3$.

We can solve $\vec{V}$ in the least squares sense as

$$A^{\mathrm{T}}_{3\times N}A_{N\times 3}\vec{V}_{3\times 1} = A^{\mathrm{T}}_{3\times N}B_{N\times 1}, \tag{13}$$

where $A^{\mathrm{T}}A$ is a 3×3 symmetric real matrix (all eigenvalues are real and positive).

The system can be solved if and only if $A^{\mathrm{T}}A$ can be reliably inverted, i.e., $A^{\mathrm{T}}A$ is nonsingular. The solution is

$$\vec{V} = \left[A^{\mathrm{T}}A\right]^{-1}A^{\mathrm{T}}B. \tag{14}$$

The eigenvalues ($\lambda_0 \le \lambda_1 \le \lambda_2$) and their corresponding eigenvectors ($\hat{e_0}$, $\hat{e_1}$ and $\hat{e_2}$) can be computed from the 3×3 symmetric least squares integration matrix $A^{\mathrm{T}}A$. When λ_0, λ_1, $\lambda_2 > \tau$, we can reliably recover a least squares 3D full velocity $\vec{V}$; otherwise, we assume there is no reliable 3D velocity there. Line normal and plane normal velocities (Barron and Spies, 2000; Spies et al., 2000a) could be defined and computed; however, they would seem to provide no useful purpose for 3D storm tracking.

Lastly, we note that theoretically, we could compute full 3D velocity from neighbouring radial velocities at adjacent time intervals as well as at one time as we show in this article. Problems that arise in this case include the fact that storms are nonrigid deformable objects (and hence the 3D motion constraint equation does not hold) and that the sampling rate of one Doppler dataset every 10 min leads to aliased data.

VI. LEAST SQUARES REGULARIZED FLOW FROM RADIAL VELOCITIES

To constrain our regularization to give a smooth full velocity field close to the true full velocity, we use the computed least squares flow as a third consistency constraint in the regularization:

$$\iiint \underbrace{(\vec{V}\cdot\hat{r} - V_r)^2}_{\text{Motion Constraint Equation}} + \alpha^2\Big(\underbrace{U_X^2 + U_Y^2 + U_Z^2 + V_X^2 + V_Y^2 + V_Z^2 + W_X^2 + W_Y^2 + W_Z^2}_{\text{Smoothness Constraint}}\Big) + \beta^2\Big(\underbrace{(U - U_{ls})^2 + (V - V_{ls})^2 + (W - W_{ls})^2}_{\text{Least Squares Velocity Consistency Constraint}}\Big)\,\partial X\,\partial Y\,\partial Z, \tag{15}$$

where $\vec{V}_{ls} = (U_{ls}, V_{ls}, W_{ls})$ is computed least squares 3D velocity. The first two constraints enforce 3D Horn and Schunck-like constraints with respect to the 3D motion constraint equation and global smoothness in the 3D velocity field. The third constraint requires that the difference between the least squares velocities and the regularized velocities be minimal. This ensures we obtain a smooth flow field, rather than the radial velocity field, as the computed velocity field. The idea here is to compute a smooth regularized velocity compatible with the local least squares velocities. α and β are Lagrange multipliers that specify the relative importance of the various constraints. On the basis of trial and error, we use $\alpha = 5.0$ and $\beta = 1.0$ to obtain the results in this article.

Since $\nabla^2 U = U_{XX} + U_{YY} + U_{ZZ}$, $\nabla^2 V = V_{XX} + V_{YY} + V_{ZZ}$ and $\nabla^2 W = W_{XX} + W_{YY} + W_{ZZ}$, after expansion of $\vec{V}\cdot\hat{r}$ as $Ur_1 + Vr_2 + Wr_3$, we can rewrite the Euler–Lagrange equations that minimize this functional as

$$\begin{aligned}(Ur_1 + Vr_2 + Wr_3)r_1 + \beta^2 U &= \alpha^2\nabla^2 U + \beta^2 U_{ls} + V_r r_1,\\ (Ur_1 + Vr_2 + Wr_3)r_2 + \beta^2 V &= \alpha^2\nabla^2 V + \beta^2 V_{ls} + V_r r_2,\\ (Ur_1 + Vr_2 + Wr_3)r_3 + \beta^2 W &= \alpha^2\nabla^2 W + \beta^2 W_{ls} + V_r r_3.\end{aligned}$$

The approximations $\nabla^2 U \approx \bar{U} - U$, $\nabla^2 V \approx \bar{V} - V$ and $\nabla^2 W \approx \bar{W} - W$ let us rewrite the Euler-Lagrange equations in matrix form as

$$A\begin{bmatrix} U \\ V \\ W \end{bmatrix} = \begin{bmatrix} (\alpha^2\bar{U} + \beta^2 U_{ls} + V_r r_1) \\ (\alpha^2\bar{V} + \beta^2 V_{ls} + V_r r_2) \\ (\alpha^2\bar{W} + \beta^2 W_{ls} + V_r r_3) \end{bmatrix}, \tag{16}$$

where

$$A = \begin{bmatrix} (r_1^2 + \alpha^2 + \beta^2) & (r_1 r_2) & (r_1 r_3) \\ (r_1 r_2) & (r_2^2 + \alpha^2 + \beta^2) & (r_2 r_3) \\ (r_1 r_3) & (r_2 r_3) & (r_3^2 + \alpha^2 + \beta^2) \end{bmatrix}. \quad (17)$$

Finally, the Gauss Seidel iterative equations can be written as

$$\begin{bmatrix} U^{n+1} \\ V^{n+1} \\ W^{n+1} \end{bmatrix} = A^{-1} \begin{bmatrix} (\alpha^2 \bar{U}^n + \beta^2 U_{\mathrm{ls}} + V_{\mathrm{r}} r_1) \\ (\alpha^2 \bar{V}^n + \beta^2 V_{\mathrm{ls}} + V_{\mathrm{r}} r_2) \\ (\alpha^2 \bar{W}^n + \beta^2 W_{\mathrm{ls}} + V_{\mathrm{r}} r_3) \end{bmatrix}. \quad (18)$$

We perform this iteration until $\|\vec{V}^{n+1} - \vec{V}^n\|_2 < 0.001$. Typically, we need about 120 iterations to satisfy this condition.

VII. PROJECTION OF 3D FLOW

We project 3D velocity vectors onto a 2D image plane for display purposes. A 3D point $\vec{P}(x, y, z)$ that moves with full velocity $\vec{V}$ will reach $\vec{P}'(x', y', z')$ after time t:

$$\begin{aligned} \vec{P}' &= \vec{P} + \vec{V}_{\mathrm{t}} \\ &= (X, Y, Z) + (V_X t, V_Y t, V_Z t) \\ &= (X + V_X t, Y + V_Y t, Z + V_Z t). \end{aligned} \quad (19)$$

Then, we can project $\vec{P}$ and $\vec{P}'$ onto a 2D XY image plane at $\vec{p}$ and $\vec{p}'$ respectively using perspective projection:

$$\vec{p} = \left(\frac{fX}{f+Z}, \frac{fY}{f+Z} \right), \quad (20)$$

$$\vec{p}' = \left(\frac{f(X + V_X t)}{f + (Z + V_Z t)}, \frac{f(Y + V_Y t)}{f + (Z + V_Z t)} \right), \quad (21)$$

where $\vec{v} \approx \vec{p}' - \vec{p}$ is the 2D projection velocity of $\vec{V}$·f is the focal length of our virtual camera, which we arbitrarily set to obtain "nice" looking images.

VIII. SYNTHETIC DOPPLER VELOCITY RESULTS

To assess the accuracy of the method, we applied our approach to synthetic radial velocities. The correctness of the approaches and their implementation can be tested by comparing the true and the estimated velocity field on the surfaces of 15 cones, which have the same coordinates and elevation angles as real Doppler radar rays.

The experimental process for the creation and analysis of synthetic Doppler data has four steps:

1. Set artificial constant full velocity for each point on the cones. The two examples we choose here are full velocities $\vec{V}_1 = (20,0,0)$ and $\vec{V}_2 = (12,–16,20)$.
2. Compute the artificial radial velocity for each point using the constant full velocities. Since the radial velocity at some atmospheric point can be viewed as the projection of the full velocity at that point along its radial direction, we have

$$V_{\mathrm{r}} = \vec{V} \cdot \hat{r} = (V_X, V_Y, V_Z) \cdot (r_X, r_Y, r_Z) = \left(\frac{V_X X}{R}, \frac{V_Y Y}{R}, \frac{V_Z Z}{R} \right) \quad (22)$$

   where $\vec{V} = (V_X, V_Y, V_Z)$ is the full velocity and $\hat{r} = (r_X, r_Y, r_Z)$ is the unit vector that indicates the direction of the radial velocity. $\hat{r}$ is computed from the Cartesian coordinates, (X, Y, Z), of the atmospheric point. The coordinate magnitude is given by $R = \sqrt{X^2 + Y^2 + Z^2}$ and is used for vector normalization.
3. Apply the least squares method to the step two radial velocities to estimate the 3D full velocity flow field.
4. Quantitatively compare the estimated velocities and the true constant velocities at each point.

A. The Normal Distribution and $N(0, \sigma^2)$ Gaussian Noise. We add random mean zero Gaussian noise to the artificial radial velocities, where the first argument of $N(0, \sigma^2)$ is the mean and σ^2 is the variance of the distribution. The method we use to generate normally distributed Gaussian noise is by Teichroew (Maisel and Gnugnoli, 1972). The procedure begins with the generation of 12 random numbers from a uniform distribution, $r_1, r_2, \ldots, r_{12} \in [0,1]$. Then we compute

$$R = \frac{\left(\sum_{i=1}^{12} r_i \right) - 6}{4} \quad (23)$$

Finally, we compute

$$x = C_1 R^9 + C_2 R^7 + C_3 R^5 + C_4 R^3 + C_5 R, \quad (24)$$

where $C_1 = 0.029899776$, $C_2 = 0.008355968$, $C_3 = 0.076542912$, $C_4 = 0.252408784$, and $C_5 = 3.949846138$. The number x has the desired standard normal distribution $N(0,1)$.

After the generation of standardised normal numbers x, we can generate numbers with any normal distributions by a simple linear transformation. That is, if x is $N(0,1)$ and $X = \mu + \sigma x$, then X is in $N(\mu, \sigma^2)$. Thus, we use standard deviation σ and mean μ (usually we let $\mu = 0$) to adjust the value of x to add controlled amounts of Gaussian noise n in $N(0, \sigma^2)$ to the radial velocities.

B. Error Measurement. We evaluate the robustness of our approaches by comparing the radial velocity input error ϕ_{I} and the output error ϕ_{O} of the full velocities.

We report the input error of radial velocity after adding Gaussian noise as

$$\phi_{\mathrm{I}} = \frac{\|\vec{n}\|_2}{\|\vec{V}_{\mathrm{t}}\|_2} \times 100\%, \quad (25)$$

where $\vec{V}_{\mathrm{t}}$ and $\vec{n}$ are $3N$ vectors, consisting of the N triplets making up the true (correct) radial velocities and the errors added to each component of those radial velocities. We compute, $\vec{n}_i$, the ith velocity noise vector as

$$\vec{n}_i = \left(\frac{n_{iX} X_i}{\sqrt{X_i^2 + Y_i^2 + Z_i^2}}, \frac{n_{iY} Y_i}{\sqrt{X_i^2 + Y_i^2 + Z_i^2}}, \frac{n_{iZ} Z_i}{\sqrt{X_i^2 + Y_i^2 + Z_i^2}} \right), \quad (26)$$

where the ith radial velocity is measured at (X_i, Y_i, Z_i) and $\vec{n}_i = (n_{iX}, n_{iY}, n_{iZ})$ is the Gaussian noise triplet added to the three components of the ith radial velocity. For output error, we report:

$$\phi_{\mathrm{O}} = \frac{\|\vec{V}_{\mathrm{t}} - \vec{V}_{\mathrm{e}}\|_2}{\|\vec{V}_{\mathrm{t}}\|_2} \times 100\%, \quad (27)$$

where $\vec{V}_{\mathrm{t}}$ is the true $3N$ 3D full velocity vector and $\vec{V}_{\mathrm{e}}$ is the $3N$ estimated 3D full velocity vector.

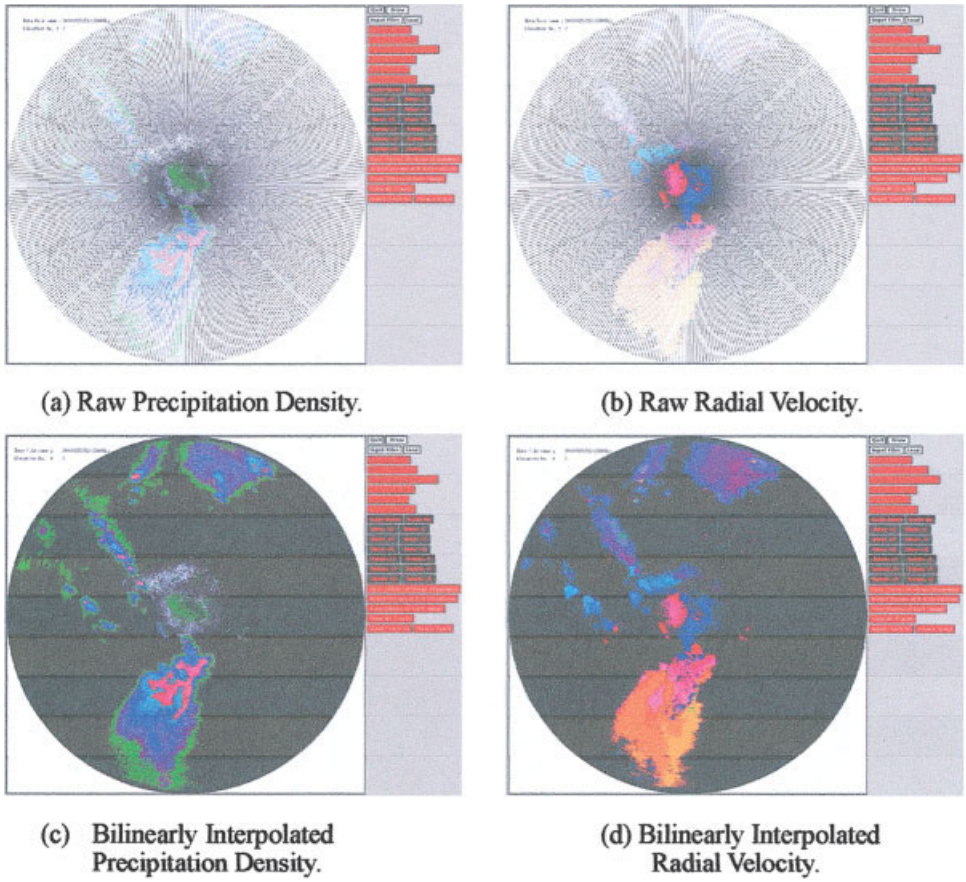


Figure 2. Real Doppler radar data obtained at time 199909161510 at elevation level 2. (a) and (c) Precipitation density, and (b) and (d) radial velocity.

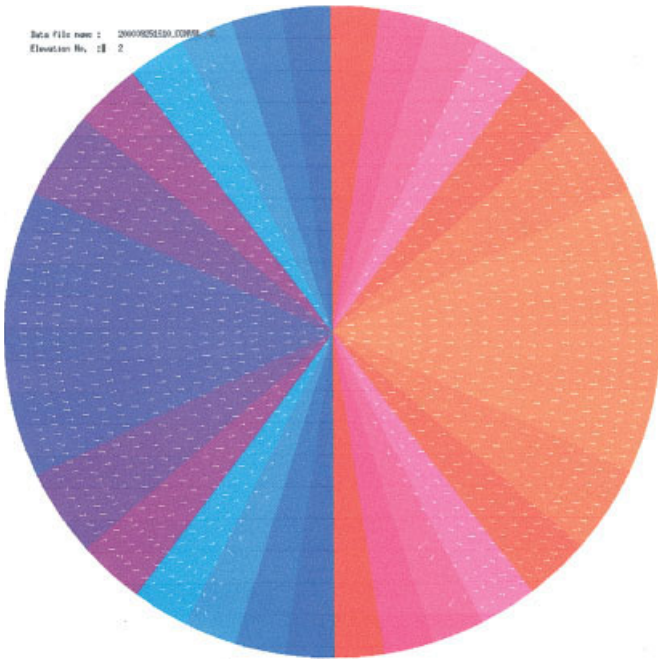


Figure 3. Visualization of least squares full velocities for noisy synthetic radial velocities with 5.76% random Gaussian noise ($\sigma = 1.0$) using the least squares method and 66.7% density. The true artificial full velocities is $\vec{V}_1 = (20,0,0)$.

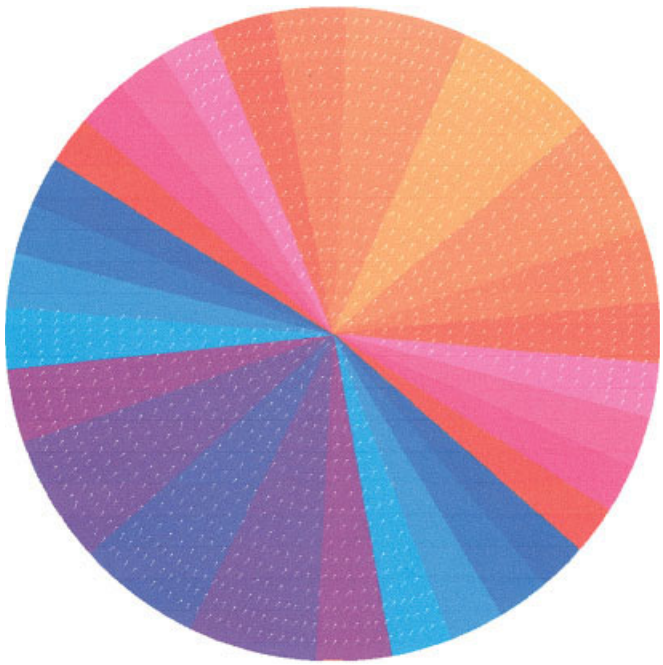


Figure 4. Visualization of least squares full velocities for noisy synthetic radial velocities with 5.68% random Gaussian noise and 66.7% density. The true artificial full velocity is $\vec{V}_1 = (12,-16,20)$.

Table I. Input error (ϕ_I) and output error (ϕ_O) for LS and R+LS 3D velocity measurements for the synthetic dataset with true full velocities $\vec{V}_1 = (20, 0, 0)$.[a]

σ	Input Error ϕ_I (%)	LS $\phi_{O\ LS}$ (%)	R+LS $\phi_{O\ R+LS}$ (%)
0.0	0.0	0.078	0.033
0.1	0.58	1.316	0.038
0.25	1.44	3.603	0.059
0.5	2.87	7.834	0.109
1.0	5.76	15.559	0.252

[a] Controlled amounts of random Gaussian noise are added on radial velocities.

Although all points in a "neighbourhood" have the same full velocity, there are some points that have no radial velocities, because the full velocity is perpendicular to the radial rays, yielding velocity projections along the ray of 0. Full velocity information cannot be recovered at these locations. Taking the velocity calculation of these points into account makes the output error calculations unreasonably high. To solve this problem, we threshold the radial velocities to get rid of the ones which are too small to yield good full velocities. In our calculation, we only consider radial velocities >10, yielding a constant density* of computed full velocities of 66.7%. Figures 3 and 4 show the computed full velocity fields for $\vec{V}_1$ and $\vec{V}_2$ with 5.76% and 5.68% random Gaussian noise (standard deviation of 1.0). As mentioned earlier, the shades are the same as those used by the Kurnell radar station to indicate radial velocity magnitude and direction (towards or away from the radar). The average magnitude and angle error for the two motions are given in Tables I and II. Figure 5 shows the computed full velocity using regularization, using the least squares constraint. In another article (Chen et al., 2001b), we show that global regularization applied on these least squares velocities can reduce the average magnitude and angle error to $<1\%$ and 1° respectively.

Full velocities only at locations where the radial velocity magnitude is ≥ 5 are computed. Typically, un-aliased radial velocities range from -40 to $+40$. We also reject full velocity calculations wherein the condition number of the least squares integration matrix is $\geq$1,000,000. Typically, these condition numbers are $<$30,000, but one did reach a value of 80,000,000 (an obvious outlier). Large condition numbers usually mean the radial velocities are at a motion discontinuity, i.e., some radial velocities are negative while others are positive.

IX. REAL DOPPLER VELOCITY RESULTS

For the least squares approach, to solve full velocity $\vec{V}$ at a point, we selected a small local neighbourhood around the point. Trial and error led us to the observation that $7 \times 7 \times 7$ neighbourhoods gave the best results. We also examined the effect of smoothing the radial velocity data before the least squares computation. The average value of the radial velocities in the $7 \times 7 \times 7$ neighbourhood of an atmospheric point, instead of the original radial velocity at that point, was used to do the computation. Figures 6 and 7 illustrate computed full velocities using unblurred and blurred radial velocities from the same dataset. The flow fields are smoother than those without blurring shown in Figure 8. These results show that smoothing the data before the velocity computation stage and increasing the neighbourhood size produce better results. Figure 9 shows the computed full velocities using regularization with the least squares constraints. Even with no smoothing of the radial velocities the computed full velocity field is smooth. Finally, Figure 10 shows the flows computed at time 200009251640 (2000, September 25, 16:40 hrs) for elevations 2–5, while Figure 11 shows the flows for the same elevations computed 40 min later at time 200009251700 (2000, September 25, 17:00 hrs). All the flows at 200009251640 are in the directions the Doppler clusters have moved to at 200009251700. Numerous other flow calculation examples are available (Chen, 2001).

A. 3D Velocity Validation. In other work (Qiu, 2001; Qiu et al., 2001; Mercer et al., 2002) we have hypothesized and tracked 3D weather storms in Doppler precipitation data. Since these storms are deforming rapidly over time, we use the notion of a "fuzzy" point (actually a 3D ellipsoid) to represent the uncertainty in the position of a storm's center of mass. We track these fuzzy points over time using a fuzzy algebra (Mercer et al., 2002) and the relaxation labelling framework we presented earlier (Cheng et al., 1998). To evaluate the quality of the computed velocities in the real data, we choose the velocity closest to a storm's fuzzy center as that storm's velocity, project that ellipsoid onto the next dataset and then compute the overlap between the projected and computed ellipsoids at the second time instance. Our conjecture is that the greater the overlap, the better the computed velocity is likely to be.

B. Intersection Volume of Two Ellipsoids. To calculate the intersection volume of two fuzzy storm ellipsoids (the predicted ellipsoid and the actual fuzzy storm ellipsoid), first, we need to obtain the predicted fuzzy storm ellipsoid center $SC'(C'_x, C'_y, C'_z)$ and radii r'_x, r'_y and r'_z as

$$C'_x = C_{x1} + U \times t \times 0.06 \tag{28}$$

$$C'_y = C_{y1} + V \times t \times 0.06 \tag{29}$$

$$C'_z = C_{z1} + W \times t \times 0.06, \tag{30}$$

where (C_{x1}, C_{y1}, C_{z1}) is the center of the initial fuzzy storm and (C'_x, C'_y, C'_z) is the center of the predicted fuzzy storm, both after time interval t (in minutes); the 0.06 factor comes from the conversion of time into seconds from minutes and velocity (displacement) into kilometers from meters. It should be noted that an assumption that the size and shape of the hypothesized fuzzy storm remain

Table II. Input error (ϕ_I) and output error (ϕ_O) for LS and R+LS 3D velocity measurements for the synthetic dataset with true full velocities $\vec{V}_2 = (20,-16,20)$.[a]

σ	Input Error ϕ_I (%)	LS $\phi_{O\ LS}$ (%)	R+LS $\phi_{O\ R+LS}$ (%)
0.0	0.0	0.052	0.033
0.1	0.57	0.969	0.036
0.25	1.42	2.635	0.048
0.5	2.85	5.981	0.083
1.0	5.68	12.823	0.185

[a] Controlled amounts of random Gaussian noise are added on radial velocities.

* The percentage of locations with velocities after thresholding.

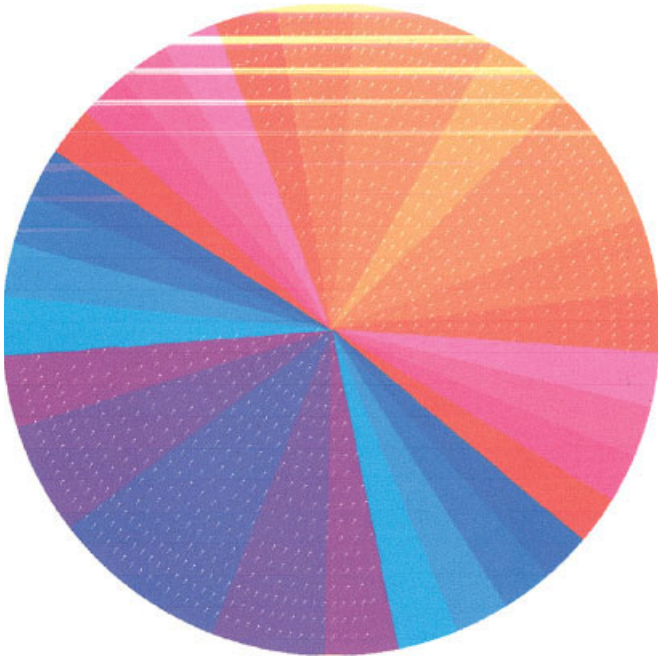


Figure 5. Visualization of the computed full Doppler velocities using regularization with the least squares velocity constraint from artificial radial velocities with 5.68% random Gaussian noise and 61.1% density. The regularization was initialized with velocities of (0,0,0). The true constant full velocity, $\vec{V}$, is $(12,-16,20)$.

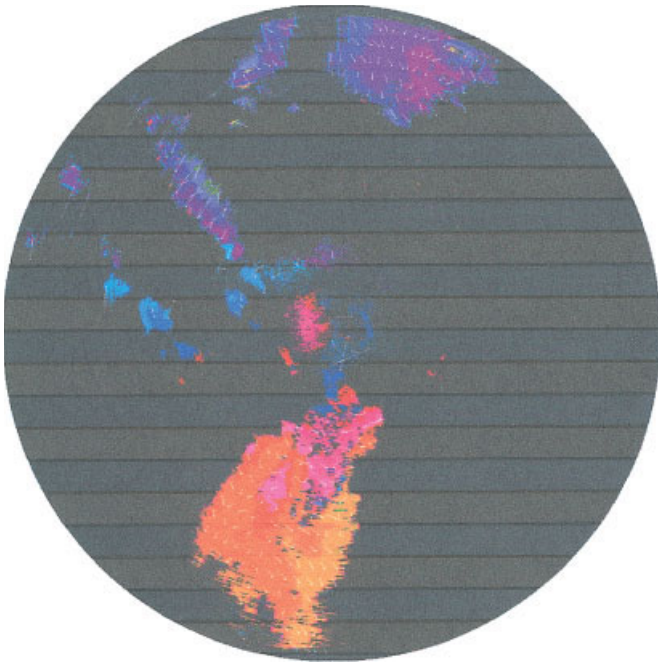


Figure 6. Computed full velocity field at elevation 2 from real Doppler radial velocity dataset 200009251510 in a least squares framework using $7 \times 7 \times 7$ neighbourhoods without smoothing.

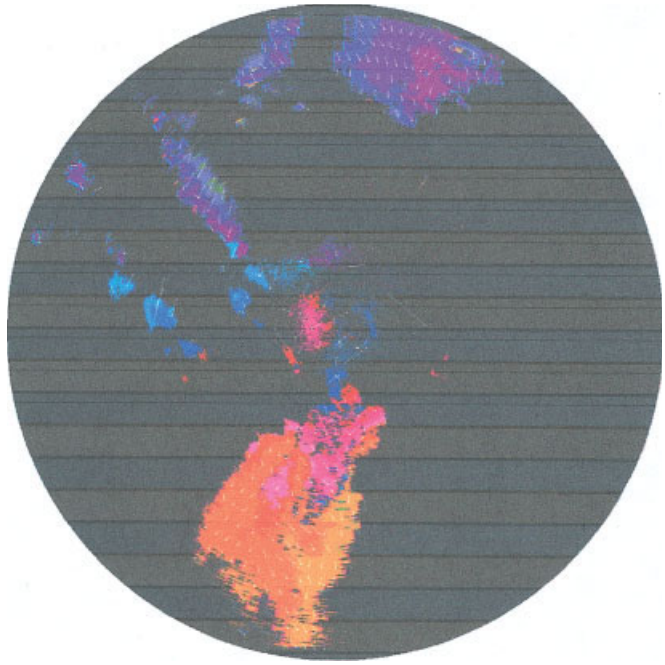


Figure 7. Computed full velocity field at elevation 2 from real Doppler radial velocity dataset 200009251510 in a least squares framework using $7 \times 7 \times 7$ neighbourhood with smoothing before the least squares computation stage.

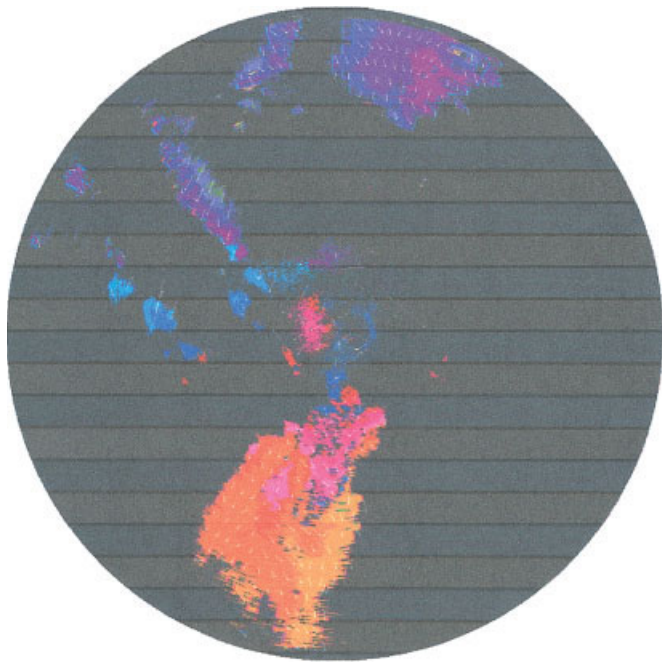


Figure 8. Computed full 3D velocity field at elevation 2 from the real Doppler radial velocity dataset at time 200009251510 in a least squares framework with a $7 \times 7 \times 7$ neighbourhood with pre-smoothing before the computation.

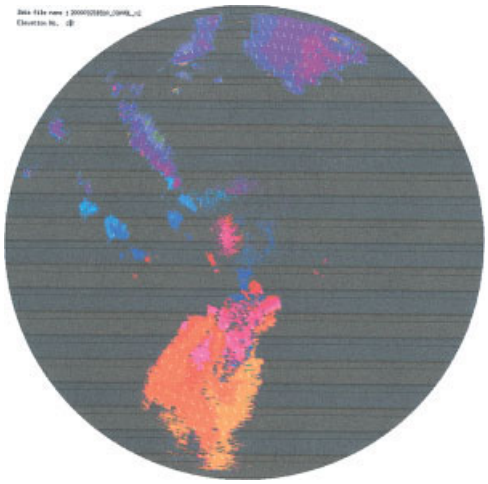


Figure 9. Computed full 3D velocity field at elevation 2 from the real Doppler radial velocity dataset at time 200009251510 using global regularization with the least squares velocity constraint.

constant within the time interval t is made in the computation of the predicted future positions of fuzzy storms (in general this is only approximately true).

Table III. Velocity intersection values, f_v, of the predicted and actual fuzzy ellipsoidal storms.

Image_1–Image_2	Volume Intersection (%)
199909161310-1320	87.62
199909161320-1330	90.17
199909161330-1340	95.00

Second, we use the equation of an ellipsoid to determine whether the points in the predicted fuzzy storm are inside the actual fuzzy storm (or vice versa). So if (X, Y, Z) is the center of a voxel that satisfies

$$\frac{(X-C_x)^2}{r_x^2}+\frac{(Y-C_y)^2}{r_y^2}+\frac{(Z-C_z)^2}{r_z^2}\le 1 \qquad (31)$$

for both the actual and predicted ellipsoids, we count it as a part of the intersection. A closed-form intersection would be desirable but is an open area of research (for example, in 3D video game playing

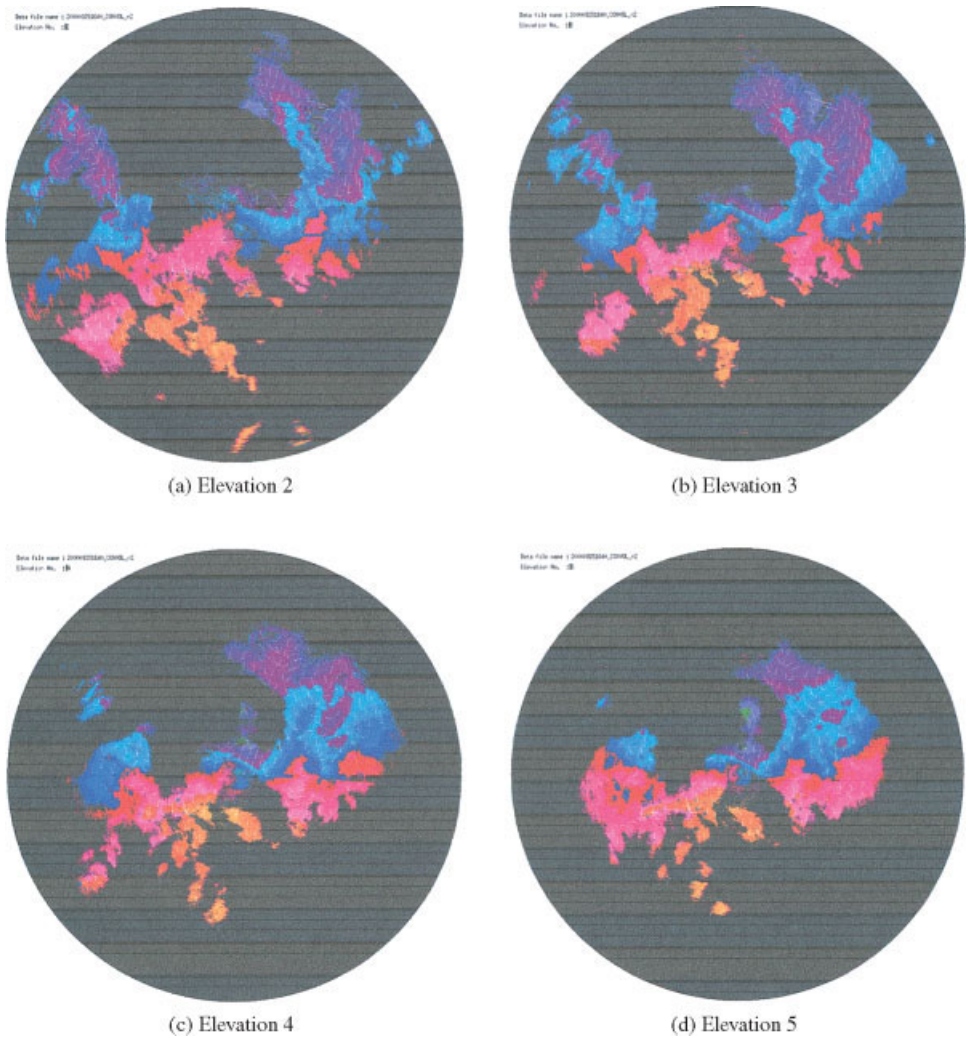
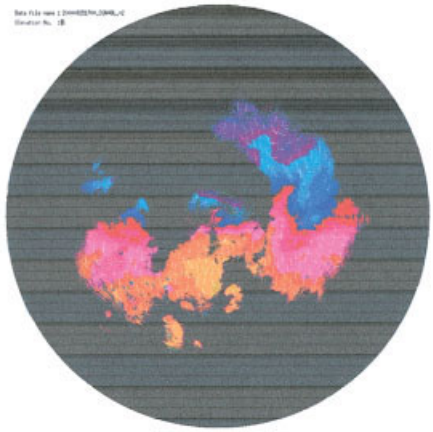


Figure 10. The least squares full velocity flow fields at elevation level 2 to 5 estimated from real radial velocity dataset obtained at time 200009251640.

(a) Elevation 2

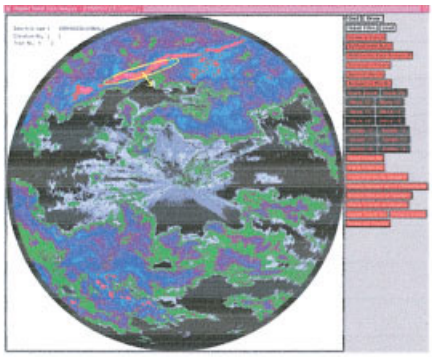
(b) Elevation 3

(c) Elevation 4

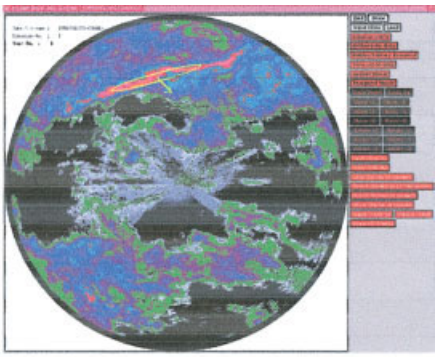


(d) Elevation 5

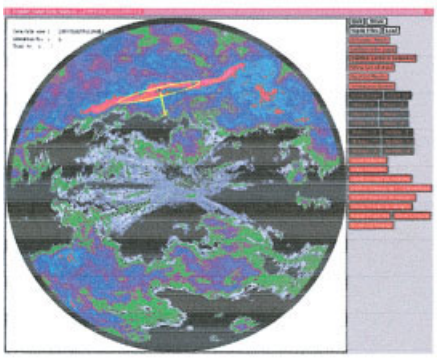
Figure 11. The least squares full velocity flow fields at elevation levels 2 to 5 estimated from real radial velocity dataset obtained at time 200009251700.



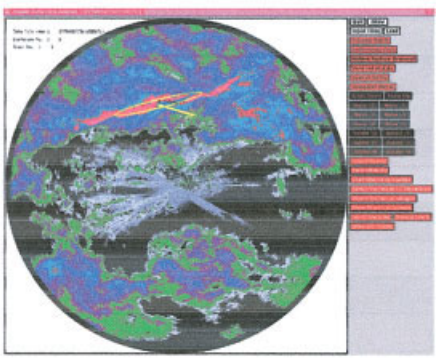
(a) Elevation 1



(b) Elevation 1



(c) Elevation 1



(d) Elevation 1

Figure 12. Velocity at the center of mass of the second storm in the reflectivity images: (a) 199909161310, (b) 199909161320, (c) 199909161330 and (d) 199909161340.

software[†]). After checking if every voxel is in an ellipsoid, we can use the count of voxels in both actual and predicted storms as an estimate of the volume of the ellipsoid intersection.

C. Volume Intersection Results. Figure 12 shows the 3D velocity values computed for the ellipsoid center of storm 2. This is the same dataset showing a large oblong storm moving from northeast to southwest (Qiu et al., 2001). The velocities are shown as vectors and can be seen to point in the direction of the storm's displacement. Table III shows the velocity intersection values, f_v, for fuzzy storms represented as ellipsoids (Qiu et al., 2001; Tang et al., 2003a,b).

X. CONCLUSIONS

We have shown quantitatively on synthetic data with controlled amounts of random Gaussian noise that we can accurately compute full 3D velocity from 3D radial velocity. We have demonstrated qualitatively (on real Doppler datasets) that we can compute good (realistic) velocity fields. We found that the radial velocities should be pre-smoothed before the least squares computation to obtain the best results. The real velocities are always in the direction of storm movement, indicating their general correctness and usefulness as a storm motion predictor. An intersection volume calculation for a number of different real storms at different times showed that computed image velocity is a good predictor for storm motion. Indeed, a velocity compatibility function is currently under design and will be used in the next version of our 3D tracking algorithm (Tang et al., 2003a).

REFERENCES

J.L. Barron, D.J. Fleet, and S.S. Beauchemin, Performance of optical flow techniques, Int J Comput Vis 12 (1994), 43–77.

J.L. Barron, R.E. Mercer, D. Cheng, and P. Joe, "Tracking 'fuzzy' storms in Doppler radar images," In Computer vision and applications handbook, B. Jähne, H. Hauβecker, and P. Geiβler (Editors), Academic Press, Boston, 1999, pp. 807–820.

J.L. Barron and H. Spies, Quantitative regularized range flow, Proc Vision Interface, May 2000, pp. 203–210.

K. Chaudhury, R. Mehrota, and C. Srinivasan, Detecting 3D flow, Proc IEEE Int Conf Robotics and Automation, May 1994, Vol. 2, pp. 1073–1078.

X. Chen, 3D velocity from Doppler radial velocity, M.Sc. Thesis, Department of Computer Science, University of Western Ontario, Ontario, 2001.

X. Chen, J.L. Barron, R.E. Mercer, and P. Joe, 3D least squares velocity from 3D Doppler radial velocity, Proc Vision Interface, Ottawa, June 2001a, pp. 56–63.

X. Chen, J.L. Barron, R.E. Mercer, and P. Joe, 3D regularized velocity from 3D Doppler radial velocity, Int Conf Image Processing, ICIP2001, Thessalonki, Greece, 2001b, Vol. 3, pp. 664–667.

D. Cheng, R.E. Mercer, J.L. Barron, and P. Joe, Tracking severe weather storms in Doppler radar images, Int J Imag Syst Technol 9 (1998), 201–213.

B.K.P. Horn and B.G. Schunck, Determining optical flow, Artif Intell 17 (1981), 185–204.

G.J. Klein and R.H. Huesman, A 3D optical approach to addition of deformable PET volumes, IEEE Nonrigid and Articulated Motion Workshop, June 1997, pp. 136–143.

G.J. Klein, B.W. Reutter, and R.H. Huesman, Non-rigid summing of grated PET via optical flow, IEEE Trans Nucl Sci 4 (1997), 1509–1512.

B.D. Lucas and T. Kanade, An iterative image-registration technique with an application to stereo vision, DARPA Image Understanding Workshop, 1981, pp. 121–130 (see also IJCAI81, pp. 674–679).

H. Maisel and G. Gnugnoli, Simulation of discrete stochastic systems, Science Research Associates, Chicago, 1972, pp. 51–52, 153–154.

R.E. Mercer, J.L. Barron, D. Cheng, and A. Bruen, Fuzzy points: Algebra and application, Pattern Recogn 35 (2002), 1153–1166.

W.H. Press, B.P. Flannery, S.A. Teckolsky, and W.T. Vetterling, Numerical recipes in C: The art of scientific computing, 2nd edition, Cambridge University Press, Cambridge, UK, 1992.

W. Qiu, Detection and tracking of 3D Doppler storms, M.Sc. Thesis, Department of Computer Science, University of Western Ontario, Ontario, 2001.

W. Qiu, R.E. Mercer, and J.L. Barron, 3D storm tracking in 3D Doppler precipitation reflectivity datasets, Irish Machine Vision and Image Processing Conf, IMVIP2001, September 2001, pp. 79–86.

E.P. Simoncelli, Design of multi-dimensional derivative filters, IEEE Int Conf Image Processing, ICIP94, 1994, Vol. 1, pp. 790–793.

S.M. Song and R.M. Leahy, Computation of 3D velocity fields from 3D cine CT images of a human heart, IEEE Trans Med Imag 10 (1991), 295–306.

S.M. Song, R.M. Leahy, D. P. Boyd, and B.H. Brundage, Determining cardiac velocity fields and intraventricular pressure distribution from a sequence of ultrafast CT cardiac images, IEEE Trans Med Imag 13 (1994), 386–397.

H. Spies, H. Haußecker, B. Jähne, and J.L. Barron, Differential range flow estimation, Symp fur Mustererkennung, DAGM1999, Springer, Bonn, September 15–17, 1999, pp. 309–316.

H. Spies, B. Jähne, and J.L. Barron, Dense range flow from depth and intensity data, Int Conf Pattern Recognition, ICPR2000, September, 2000a, Vol. 1, pp. 131–134.

H. Spies, B. Jähne, and J.L. Barron, Regularised range flow, European Conf Computer Vision, ECCV2000, June, 2000b.

H. Spies, B. Jähne, and J.L. Barron, Range flow estimation, Comput Vis Image Understand 85 (2002), 209–231.

X. Tang, J.L. Barron, R.E. Mercer, and P. Joe, Tracking weather storms using 3D Doppler radial velocity information, 13th Scandinavian Conf Image Analysis, SCIA2003, 2003a.

X. Tang, J.L. Barron, R.E. Mercer, and P. Joe, Tracking 3D Doppler weather storms using fuzzy ellipsoids, Irish Machine Vision and Image Processing Conf, IMVIP2003, 2003b, pp. 73–82.

M. Yamamoto, P. Boulanger, J. Beraldin, and M. Rioux, Direct estimation of range flow on deformable shape from a video rate range camera, IEEE Trans Pattern Anal Mach Intell 15 (1993), 82–89.

Z. Zhou, C.E. Synolakis, R.M. Leahy, and S.M. Song, Calculation of 3D internal displacement fields from 3D X-ray computer tomographic images, Proc R Soc Lond A 449 (1995), 537–554.

[†]www.magic-software.com