3D Least Squares Velocity from 3D Doppler Radial Velocity

X. Chen, J.L. Barron, R.E. Mercer
Dept. of Computer Science
University of Western Ontario
London, Ontario, Canada, N6A 5B7
{xchen,barron,mercer}@csd.uwo.ca

P. Joe
King City Radar Station
Atmospheric Environmental Services
4095 Dufferin St., Toronto, Ontario, M3H 5T4
Paul.Joe@ec.gc.ca

Abstract We present a local least squares computational approach for computing 3D velocity (3D optical flow) from 3D radial velocity. We demonstrate the performance of our algorithm 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 motion of storms in sequences of Doppler radar datasets. **Keywords:** 3D Optical Flow, 3D Doppler Full/Radial Velocity, Local Least Squares.

1 Introduction

We present an extension of our 2D Doppler storm tracking work into 3D [7, 8, 13] by using local 3D radial velocity neighbourhoods to compute local 3D velocity. Radial velocity (measured by the Doppler effect) is the component of 3D velocity along the radial vector from the radar station to some 3D moving atmospheric point. Our computation of full velocity from environmental radial velocities uses a local least squares framework, much like Lucas and Kanade [16]. An iterative regularization technique, like Horn and Schunck's [11], is presented in [5, 6]. In addition to computing 3D velocity from radial velocities, it is also possible to use densely sampled range sequences [22] to compute **range flow**, the 3D velocity of environmental points relative to a range sensor, in a similar manner [20, 21]. Note that range flow is computed with respect to a moving 3D surface rather than moving 3D volumetric data (3D radial velocity).

2D optical flow methods have recently been generalized into the 3D domain. Chaudhury et al. [4] formulated a 3D optical flow constraint, using I_x, I_y, I_z and I_t derivatives. Thus, they have a time-varying volume of intensity where all 4 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 [18, 19, 23, 15, 14, 2, 9, 10, 12].

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 equation (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. Equation (1) is one equation in two unknowns and manifests the *aperture* problem. The true velocity is some point on this line. Normal velocity (the component of image velocity normal to the local intensity structure) can be totally expressed in terms of derivative information:

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

while tangential velocity, $\vec{v}_t$ cannot, in general, be recovered. Normal velocity can be shown to be the point on the motion constraint line closest to the origin. Note that

$$\vec{v} \cdot \hat{n} = v_n, \tag{3}$$

where $\hat{n}$ is the unit normal direction vector, is another equivalent way to write equation (1), with $\vec{v}_n = v_n \hat{n}$.

To resolve the aperture problem and solve for $\vec{v}$ we need to impose an additional constraint. An example of a local constraint is to assume that locally all image velocities are the same. For example, Lucas and Kanade [16] use a least squares computation to integrate local neighbourhoods of normal image velocities into full image velocities. 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

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

where A has entries I_{xi} and I_{yi} in the i^{th} row and B has entries $-I_{ti}$ in the i^{th} row. We can perform eigenvector/eigenvalue analysis on $A^T A$ using routines in [17]. Eigenvalue ($\lambda_0 \leq \lambda_1$) and corresponding eigenvector ($\hat{e}_0$ and $\hat{e}_1$) decomposition of the symmetric matrix $A^T A$ yields least squares full image velocity, if both $\lambda_0, \lambda_1 > \tau$, or least squares normal image velocity, $\vec{v}_{ln} = \vec{v} \cdot \hat{e}_1$, if $\lambda_1 > \tau$ but $\lambda_0 \leq \tau$ [3].

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

$$I_X U + I_Y V + I_Z W + I_t = 0, \tag{5}$$

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. Equation (5) describes a 3D plane, with the true 3D velocity being a point on that plane. The aperture problem is manifested as the velocity on this plane closest to the origin (the **plane** normal velocity). If 3 distinct plane normal velocities are the result of the projection of a single 3D velocity, then these different 3D planes intersect at a unique point, which is that projected 3D velocity. If two planes intersect (at a line) we call the point on that line closest to the origin the **line** normal velocity. Plane and line normal velocities are described in full detail in Spies et al. [20, 21] The 3D motion constraint equation can be rewritten as

$$\vec{V} \cdot \hat{n} = V_n, \tag{6}$$

where $\vec{V_n} = V_n \hat{n}$ is a 3D plane normal velocity which can be described solely in terms of intensity derivatives:

$$\vec{V_n} = \frac{-(I_X, I_Y, I_Z) I_t}{||(I_X, I_Y, I_Z)||_2^2}. \tag{7}$$

Both the 2D and 3D motion constraint equations assume rigid motion amd pure translation. When velocity derivative data is computed from intensity data, lambertian shading is also assumed, meaning there are no specularities and all intensity changes are assumed to be due to motion alone. We use the 3D constraint equation here under the assumption that locally these first two assumptions hold (radial velocity obviously has no intensity component). Although Doppler data is highly deformable, the radial velocity data is effectively instantaneously sampled [1], making these first two assumptions valid.

2 Doppler Data

Our 3D Doppler data consists of datasets sampled at 10 minute intervals. Each data set is comprised 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. 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 but also to view all the elevations at one time, both in movie mode. As well, all the elevation data at one time can be simultaneously viewed (using Z buffer hidden surface removal) and manipulated by 3D affine transformations [5]. Our tool also performs bilinear interpolation to allow graphically pleasing views of the data. Figure 1 illustrates the 3D structure of Doppler radar datasets. Figure 2 shows the

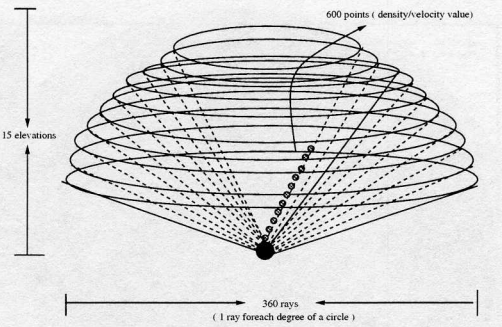
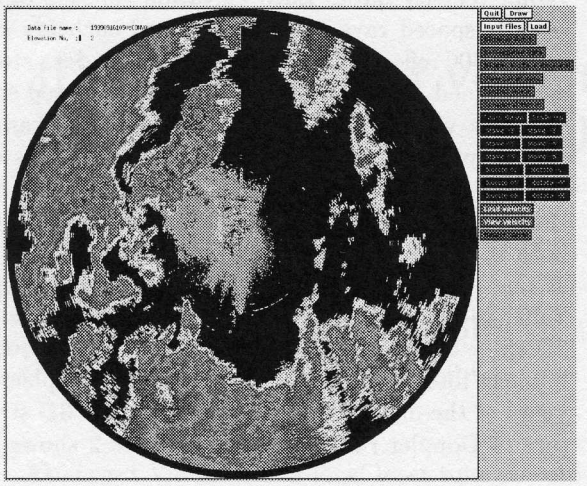


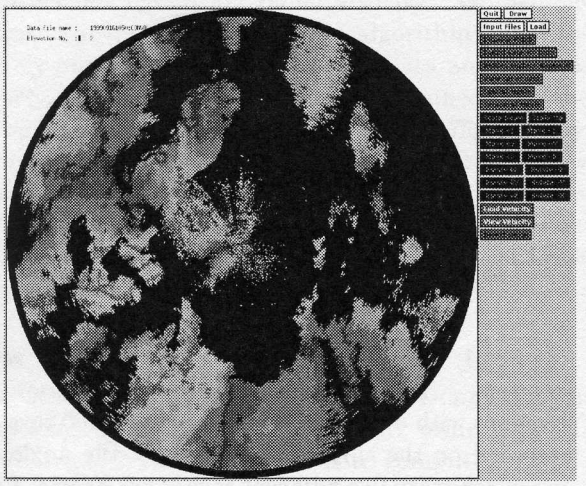
Figure 1: *The 3D structure of Doppler radar datasets.* The scanning surfaces of radar rays are 15 cones with different elevation angles. 360 rays are recorded on the surface of each cone; the angle between arbitrary two adjacent rays is *one degree*. Each ray is divided into equal parts by 600 points which contains the reflectivity and the radial velocity of atmospheric water precipitation. Each ray point location is described in 3D spherical polar coordinates.

bilinearly interpolated reflectivity and radial velocity data for time 199909161510 (1990, September 16, 15:10) at elevation 2 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 these and all other radar images in this paper.

Currently this data is stored in a three-dimensional array which 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.



(a) Precipitation Density.



(b) Radial Velocity.

Figure 2: *Real Doppler radar data obtained at time 199909161510 at elevation level 2.* (a) Precipitation Density and (b) Radial Velocity.

3 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 $\hat{r}$, i.e. $\vec{V} \cdot \hat{r} = V_r$, to obtain:

$$Ur_X + Vr_Y + Wr_Z = V_r, \tag{8}$$

which is one equation in three unknowns. Essentially, this is the 3D motion constraint equation for 3D optical flow, i.e. equation (6), but with radial velocity replacing plane normal velocity.

To solve for $\vec{V}$ 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 is smoothed. For each point, we calculate the average radial velocity in its $7 \times 7 \times 7$ neighbourhood, then replace the original radial velocity at that point with its average value. Experimental results here and in a thesis [5] show smoothing the data before the computation and using large size neighbourhoods for least squares integration produce better results. One should note that this box filtering approach is sound as, for all effective purposes, the data is uniformly sampled. Two adjacent rays are oriented differently by 1 degree (or less in the Z dimension for the lower elevations), yielding radial velocity data that is very close to being uniformly sampled.

Since the neighbourhood is small compared to 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 and 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

$$\begin{bmatrix} r_{X_1} & r_{Y_1} & r_{Z_1} \\ r_{X_2} & r_{Y_2} & r_{Z_2} \\ \cdots & \cdots & \cdots \\ \cdots & \cdots & \cdots \\ r_{X_N} & r_{Y_N} & r_{Z_N} \end{bmatrix} \begin{bmatrix} U \\ V \\ W \end{bmatrix} = \begin{bmatrix} V_{r1} \\ V_{r2} \\ \cdots \\ \cdots \\ V_{rN} \end{bmatrix}, \tag{9}$$

which can be written as:

$$A_{N\times 3}\vec{V} = B_{N\times 1}, \tag{10}$$

where A has entries r_{X_i}, r_{Y_i} and r_{Z_i} in the i^{th} row, B has entry V_{r_i} in the i^{th} 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. At these locations most of the velocity information is either tangential to the radial direction and/or the velocity data is aliased, meaning that the use of the 3D motion constraint equation to recover the 3D velocities is likely fail. A has the size $N \times 3$ when there is no thresholding. An alternate approach to thresholding suggests itself: threshold radial velocities where the precipitation density data is low. However, because the precipitation and radial velocity data are sampled at slightly different times, they do not perfectly

correlate. Nevertheless, we will investigate this possibility further in future work.

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

$$A^T_{N\times 3} A_{N\times 3} \vec{V} = A^T_{N\times 3} B_{N\times 1}, \quad (11)$$

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

The system can be solved if and only if $A^T A$ can be reliably inverted, i.e. $A^T A$ is non-singular. The solution is:

$$\vec{V} = [A^T A]^{-1} A^T B. \quad (12)$$

The eigenvalues ($\lambda_0 \leq \lambda_1 \leq \lambda_2$) 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 $A^T A$. When $\lambda_0, \lambda_1, \lambda_2 > \tau$, 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 [20, 21] can possibly then be defined here and computed, however they 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 paper. Problems that arise in this case include the fact that storms are non-rigid deformable objects (and hence the 3D motion constraint equation would not hold) and that the sampling rate of one Doppler dataset every 10 minutes leads to aliased data.

4 Projection of 3D Flow

3D velocity vectors are projected onto a 2D image plane for display purposes.

A 3D point $\vec{P}(x, y, z)$ which moves with full velocity $\vec{V}$ will reach $\vec{P}'(x', y', z')$ after time t:

$$\begin{aligned} \vec{P}' &= \vec{P} + \vec{V}t \\ &= (X, Y, Z) + (V_X t, V_Y t, V_Z t) \\ &= (X + V_X t, Y + V_Y t, Z + V_Z t). \end{aligned} \quad (13)$$

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:

$$\vec{p} = (\frac{fX}{f+Z}, \frac{fY}{f+Z}), \quad (14)$$

$$\vec{p}\,' = (\frac{f(X+V_X t)}{f+(Z+V_Z t)}, \frac{f(Y+V_Y t)}{f+(Z+V_Z t)}), \quad (15)$$

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.

5 Synthetic Doppler Velocity Results

In assess the accuracy of the method, we applied our approach to synthetic radial velocities. We generate synthetic data by projecting 3D velocity along the same radial lines as the Doppler radar uses. Full velocity was then recovered in our least squares framework using this synthetic data with controlled noise added to it. 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.

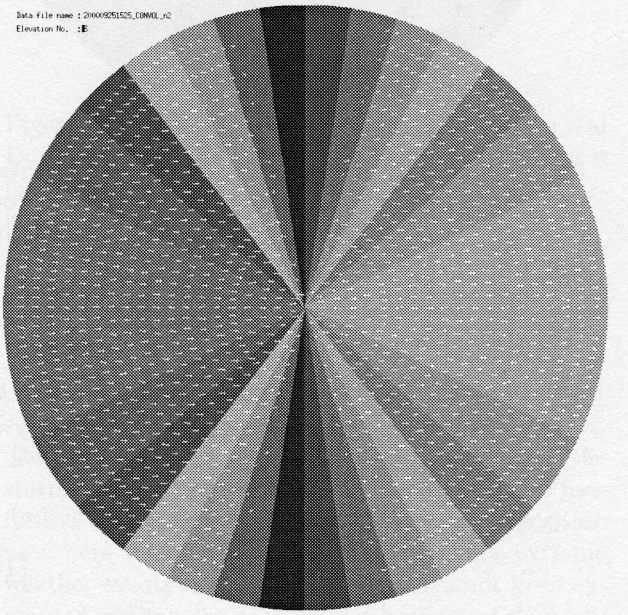


Figure 3: Visualization of least squares full velocities for noisy synthetic radial velocities with 10% random Gaussian noise ($\sigma = 1.0$) using the least squares method and 80.47% density. The true artificial full velocities is $\vec{V}_1 = (20, 0, 0)$.

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

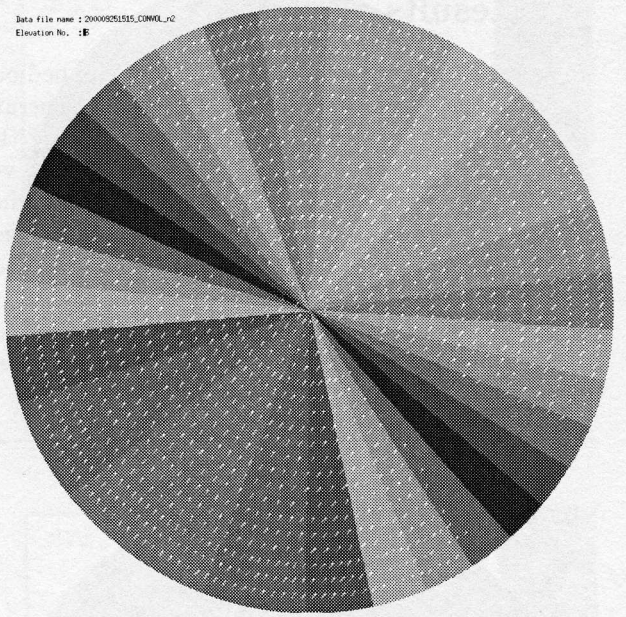


Figure 4: Visualization of least squares full velocities for noisy synthetic radial velocities with 10% random Gaussian noise and 80.29% density. The true artificial full velocity is $\vec{V}_2 = (12, -16, 20)$.

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:

$$\begin{aligned} V_r &= \vec{V} \cdot \hat{r} = (V_X, V_Y, V_Z) \cdot (r_X, r_Y, r_Z) \\ &= \frac{V_X X}{R}, \frac{V_Y Y}{R}, \frac{V_Z Z}{R} \end{aligned} \quad (16)$$

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 which 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 purposes.

3. Apply the least squares method to step 2 radial velocities velocities to estimate the 3D full velocity flow field.

4. Quantitatively compare the estimated velocities and the true constant velocities at each point.

Figures 3 and 4 show the computed full velocity fields for $\vec{V}_1$ and $\vec{V}_2$ with 10.0% random Gaussian noise (standard deviation of 1.0). As mentioned earlier, the shades (actually colour, although not reproduced here) are the same as those used by the Kurnell radar station to indicate radial velocity magnitude and direction (towards or aways from the radar). The average magnitude and angle error for the 2 motions is shown in Table 1 below. In another paper [6], we show that global regularization applied on these least squares velocities can reduce the average magnitude and angle error to less than 1% and 1° respectively.

We only compute full velocities at locations where the radial velocity magnitude is ≥ 5. Typically, un-aliased radial velocities range from -40 to $+40$. We also reject full velocity calculations where the condition number of the least squares integration matrix is $\geq$ 1,000,000. Typically, these condition numbers are less than 30,000 but one did reach a value of 80,000,000 (an obvious outlier). Large condition numbers usually mean the radial velocity are at a motion discontinuity, i.e. some radial velocities are negative while others are positive. Figure 6 illustrates computed full velocities using blurred radial velocities from the same Doppler dataset. The flow fields are smoother than those without blurring. These results show that smoothing the data before the velocity computation stage and increasing the neighbourhood size produce better results. Finally, Figure 7 show the flows computed at time 200009251640 (2000, September 25, 16:40) for elevations 2 and 5 while Figure 8 show the flows for the same two elevations computed 40 minutes later at time 200009251700 (2000, September 25, 17:00). All the flows at 200009251640 are in the directions the Doppler clusters have moved to at 200009251700. For elevation 2, the average velocity is 20m/sec or 1.2km/minute. After 20 minutes this corresponds to a predicted displacement of 24km or (after the appropriate scaling) 18 pixels. The approximate displacement of the storm between the two images is 20 pixels, which is quite close to the predicted displacement. Numerous other flow calculation examples are available [5].

6 Real Doppler Velocity Results

For the least squares approach, to solve for full velocity $\vec{V}$ at a point, we select 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

Dataset	Mag. Error	Angle Error	Density
$\vec{V}_1$ (5%)	3.171% ± 0.282%	3.284° ± 7.415°	81.49%
$\vec{V}_1$ (10%)	5.926% ± 0.621%	6.814° ± 13.7817°	80.47%
$\vec{V}_2$(5%)	3.689% ± 0.297%	2.315° ± 3.823°	81.32%
$\vec{V}_2$(10%)	6.699% ± 0.778%	4.618° ± 6.515°	80.29%

Table 1: Magnitude and angle errors for $\vec{V}_1$ and $\vec{V}_2$ for 5% and 10% random Gaussian noise ($\sigma = 1.0$). Density were obtained by thresholding all radial velocities with magnitude less than 5.0.

point, instead of the original radial velocity at that point, was used to do the computation.

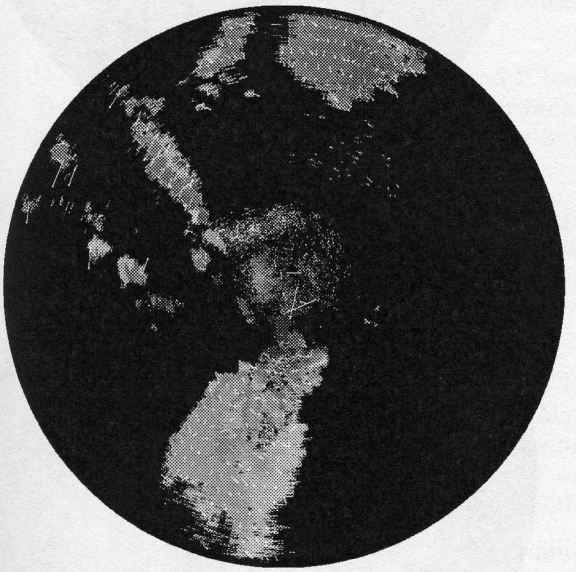


Figure 5: Computed full velocity flow field for real Doppler radial velocity dataset 200009251510 in a least squares framework using $7 \times 7 \times 7$ neighbourhoods without smoothing.

7 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 shown, qualitatively on real Doppler datasets, that we can compute good (realistic) velocity fields. We found that the radial velocities should be pre-smoothed before a least squares computation to obtain the better results. The real velocities are always in the direction of the storms, indicating their general correctness and usefulness as a storm motion predictor. The algorithm used here works for severe weather storms, extreme weather events such as tornadoes, have not been examined and may be problematic because of their relatively small sizes.

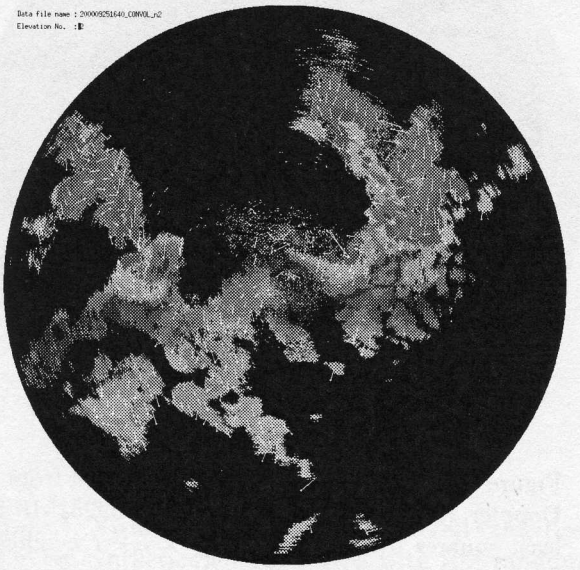
Figure 6: Computed full velocity flow field from real Doppler radial velocity dataset 200009251510 in a least squares framework using $7 \times 7 \times 7$ neighbourhood with smoothing before the least squares computation stage.

Acknowledgements: We gratefully acknowledge support from NSERC (National Science and Engineering Research Council of Canada) operating grants and an AES (Atmospheric Environmental Services) subvention.

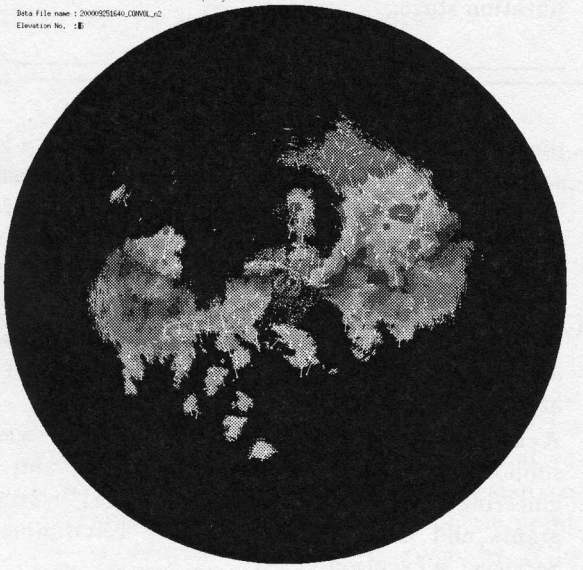
References

[1] Doppler signal processor user's manual. Sigmet Inc., MA, USA, November 1997.

[2] S.C. Amartur and H.J. Vesselle. A new approach to study cardiac motion: The optical flow of cine mr images. *Magnetic Resonance in Medicine*, 29(1):59–67, 1993.

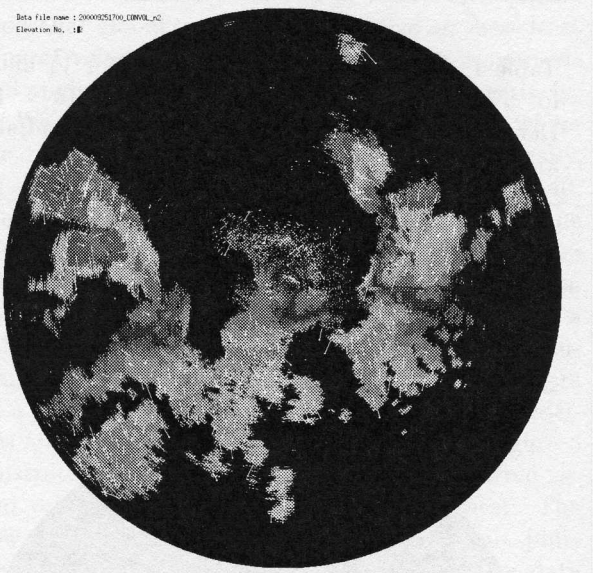


(a) Elevation 2

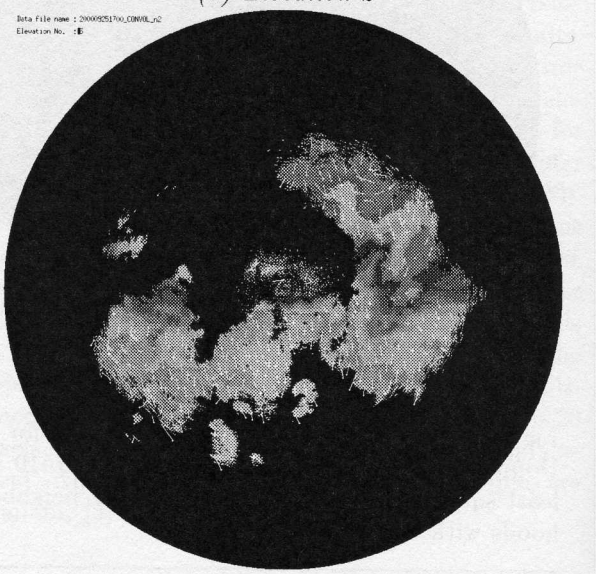


(a) Elevation 5

Figure 7: The least squares full velocity flow fields at elevation level 2 and 5 estimated from real radial velocity dataset obtained at time 200009251640.



(a) Elevation 2



(b) Elevation 5

Figure 8: The least squares full velocity flow fields at elevation level 2 and 5 estimated from real radial velocity dataset obtained at time 200009251700.

[3] J. L. Barron, D. J. Fleet, and S. S. Beauchemin. Performance of optical flow techniques. *IJCV*, 12(1):43–77, 1994.

[4] K. Chaudhury and R. Mehrota amd C. Srinivasan. Detecting 3d flow. In *Proc. IEEE Int. Conf. Robotics and Automation*, volume 2, pages 1073–1078, May 1994.

[5] X. Chen. 3d velocity from doppler radial velocity. Master's thesis, The Dept. of Computer Science, The University of Western Ontario, March 2001.

[6] X. Chen, J.L. Barron, R.E. Mercer, and P. Joe. 3d regularized velocity from 3d doppler radial velocity. In *submitted*, January 15th 2001.

[7] D. Cheng, R.E. Mercer, J.L. Barron, and P. Joe. Tracking fuzzy storm centres in doppler radar images. In *IEEE Int. Conf. on Image Processing (ICIP96)*, volume 2, pages 959–962, September 1996.

[8] D. Cheng, R.E. Mercer, J.L. Barron, and P.Joe. Tracking severe weather storms in doppler radar images. *Int. Journal of Imaging Systems and Technology*, 9:201–213, August 1998.

[9] S. N. Gupta and J. L. Prince. On div-curl regularization for motion estimation in 3d volumetric imaging. In *Int. Conf. Image Processing (ICIP96)*, volume 1, pages 929–932, September 1996.

[10] M.A. Gutierrez, M.S. Rebelo, S.S. Furuie, C.P. Melo, L. Moura, L. Avila, and J.R. Pargo. Myocardial motion estimation in gated-mri using optical flow refined with scale-space. In *Conf. on Computers in Cardiology*, pages 153–156, 1997.

[11] B. K. P. Horn and B. G. Schunck. Determining optical flow. *Artificial Intelligence*, 17:185–204, 1981.

[12] S.-H. Huang, S.-T. Wang, and J.-H. Chen. 3d motion analysis of mr imaging using optical flow method. In *Engineering in Medicine and Biology Society, 17th Annual International Conference of the IEEE*, pages 463–464, 1995.

[13] J.L.Barron, R.E. Mercer, D Cheng, and P. Joe. Tracking 'fuzzy' storms in doppler radar images. In Bernd Jähne, Horst Haußecker, and Peter Geißler, editors, *Computer Vision and Applications Handbook*, pages 807–820. Academic Press, Boston, January 1999.

[14] G.J. Klein, B.W. Reutter, and R.H. Huesman. Non-rigid summing of grated pet via optical flow. *IEEE Trans. on Nuclear Science*, 4(1):1509–1512, August 1997.

[15] Gregory J. Klien and Ronald H. Huesman. A 3d optical approach to addition of deformable pet volumes. In *IEEE Nonrigid and Articulated Motion Workshop*, pages 136–143, June 1997.

[16] B. D. Lucas and T. Kanade. An iterative image-registration technique with an application to stereo vision. In *Image Understanding Workshop*, pages 121–130. DARPA, 1981. (see also IJCAI81, pp674-679).

[17] 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, 1992.

[18] S.M. Song and R.M. Leahy. Computation of 3d velocity fields from 3d cine ct images of a human heart. *IEEE Trans. Medical Imaging*, 10(1):295–306, 1991.

[19] 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. Medical Imaging*, 13(2):386–397, 1994.

[20] H. Spies, H. Haußecker, B. Jähne, and J.L.Barron. Differential range flow estimation. In *21.Symposium fur Mustererkennung (DAGM'1999)*, pages 309–316. Springer, Bonn, September 1999.

[21] H. Spies, B. Jähne, and J.L. Barron. Regularised range flow. In *European Conference on Computer Vision (ECCV2000)*, June 2000.

[22] M. Yamamoto, P. Boulanger, J. Beraldin, and M. Rioux. Direct estimation of range flow on deformable shape from a video rate range camera. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(1):82–89, January 1993.

[23] 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(1937):537–554, 1995.