Motion Vectors in Weather Radar Images

M. Peura and H. Hohti

Finnish Meteorological Institute, Remote Sensing for Weather Applications
P.O. Box 503 (Vuorikatu 24) 00101 Helsinki, Finland

Abstract

In this paper, we discuss the extraction motion vectors in weather radar images. Our goal is to provide fast computable but still reliable nowcasting of precipitation. The problem is analogous to motion extraction and extrapolation in satellite images, especially in cases where background has been screened off by some method. Our emphasis is on optical flow algorithm, which is an alternative to autocorrelation based motion detection algorithms. For implementation, we suggest fast, sliding-window based computing techniques as well as usage of data quality information.

1 Introduction

In forecasting precipitation for the next couple of hours, direct radar data extrapolation typically is superior to numerical weather forecasting models. The processed radar images are typically at least 1000×1000 pixels large and the required maximal processing time is around a minute. These preconditions require efficient algorithms in motion extraction (and extrapolation).

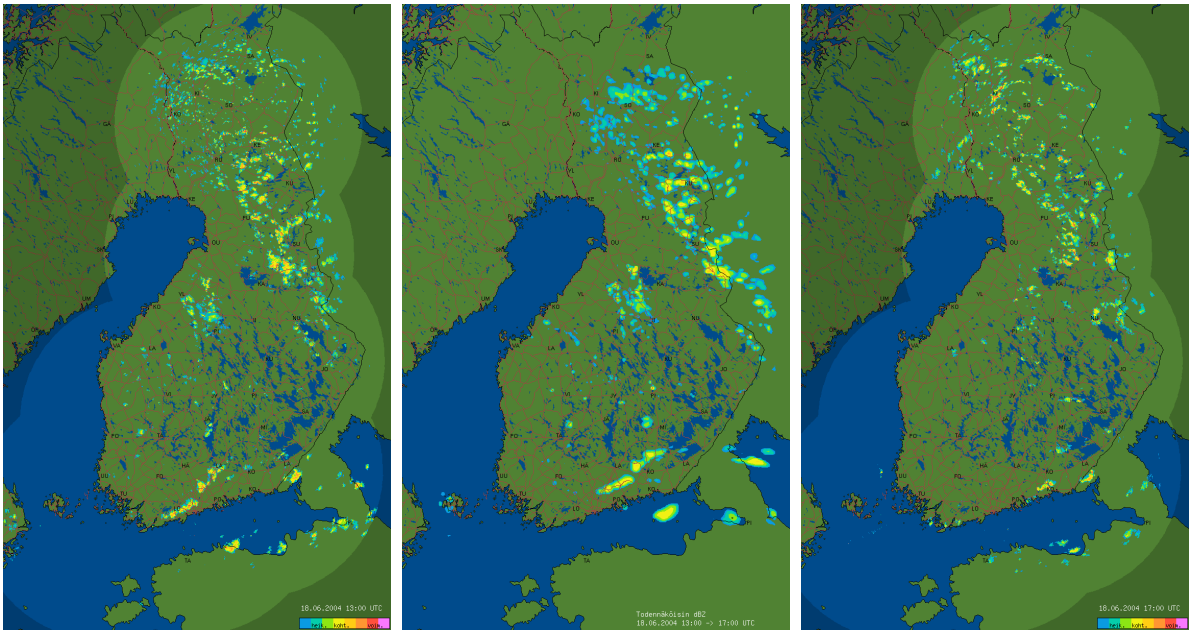
In weather radar measurements, two principal types of motion can be observed. First, there is average water droplet motion that can be observed by means of a Doppler radar network. Second, there is the motion of a precipitation area that appears in a sequence of radar images. When nowcasting precipitation, the latter type is more important.

Both of these motion types are affected by winds, but in a different manner. In the case of water droplets a practical assumption is that their motion approximates the wind and hence apply Doppler radar for measuring (radial) wind speeds. The assumption is valid especially when measuring horizontal motion of small droplets. On the other hand, in the motion of the precipitation area, the affect of winds is often overridden by atmospheric dynamics generating the precipitation. Furthermore, motion tends to diverge into different scales and phenomena. For example, areas of frontal rain, areas of (embedded) convection, and separate convective cells may have motions of their own.

Next, we briefly review the *autocorrelation* based forecasting facility applied in the FMI. In the rest of the paper we focus on the *optical flow algorithm*, first reviewing the basics and then suggesting details in the implementation.

2 Autocorrelation based techniques

The precipitation forecasting application used in FMI is based on the EUMETSAT motion vector package designed by Holmlund (2000) who also provided the modifications required by radar data. The obtained precipitation forecast is one of the most popular WWW broducts at the FMI. A sample scene is shown in Fig. 2.



Observation at 13 UTC | Forecast for 17 UTC made at 13 UTC | Observation at 17 UTC

Figure 1: Sample images from the operational precipitation forecasting facility of the FMI.

The motion detection of precipitating areas is done in four 16x16 1km pixel grids having 8 km offsets each. As a first guess, the detection is done using unfiltered reflectivity data. The half-hourly time series of first guess vector fields are used to analyze vector qualities and components and their spatial and temporal variations.

The result of the time series analysis can be used as an input to the anomaly detection and removal process. As part of this process, the vector field characteristics are used to determine the probabality of echoes belonging to ”naturally moving” precipitating ones. As a side effect, this also helps the classification of anomalous echoes having more or less chaotic behaviour from the movement detection point of view.

Finally, the best guess motion vector field of precipitating echoes can be obtained directly from the analysis of first guess fields, or it can be done separately (or even iteratively) using anomaly free data. Before using it as an input to the nowcasting process, the vector field has to be interpolated to spread the field over non-precipitating areas. This is necessary, because trajectories for up to four forecast hours has to be extended outside the precipitating areas also.

3 Optical flow — basics

In computer vision, *optical flow* is one of the standard techniques in computing motion vectors from two subsequent images (Sonka et al. 1993, J. L. Barron and Fleet 1994). The underlying idea is to explain observed temporal and spatial derivatives with a continuous motion field — a *flow*. The model is analogous to meteorogical advection models, which suggest using this approach in related imageries as well.

More formally, the flow of quantity $f = f(x, y, t)$ can be modelled as

$$\frac{df}{dt} = \frac{\partial f}{\partial t} + \frac{\partial f}{\partial x}\frac{\partial x}{\partial t} + \frac{\partial f}{\partial y}\frac{\partial y}{\partial t} = f_t + f_x u + f_y v = f_t + \nabla f \cdot \mathbf{v}. \quad (1)$$

where df/dt is the change observed in the coordinates of the flowing data and the partial derivatives indicate the changes in the rigid (image) coordinates. To illustrate, one may think about staring at a fixed position in a flow image frame and observing how f changes along a slope of the data flowing past that point: the observed change $\partial f/\partial t$ is a function of the slope steepness ∇f and the flow velocity $\mathbf{v} = [u\ v]^T$ in that point, plus a possible change in the data itself, df/dt.

Approaching practical remote sensing problems, we replace the continuous data field by digital image f and redefine $f_x = (f(x+1, y, t) - f(x-1, y, t))/2$ and $f_y = (f(x, y+1, t) - f(x, y-1, t))/2$. Likewise, we introduce difference image $f_t = f(x, y, t) - f(x, y, t-1)$. We assume further that all changes in intensity are explained by motion, that is, there are no changes in the data: $df/dt \equiv 0$. Hence we have

$$\nabla f \cdot \mathbf{v} + f_t = 0 \quad (2)$$

As this single equation contains two unknowns (u and v), there are infinitely many solutions. The technique discussed by J. L. Barron and Fleet (1994) applies a neighbourhood Ω of point (x, y) and yields an overdetermined flow equation set which leads to cost minimization of type

$$C = \sum_{\Omega} w \cdot (\nabla f \cdot \mathbf{v} + f_t)^2 \quad (3)$$

where weight w can be a function of image coordinates and/or neighborhood coordinates. In matrix form, derivation with respect to $\mathbf{v}$ yields

$$\mathbf{GW}\left[\mathbf{G}^{\mathrm{T}}\mathbf{v} + \mathbf{f_t}\right] = \begin{bmatrix} 0 \\ 0 \end{bmatrix}. \quad (4)$$

where $\mathbf{G} = [\nabla f_{(1)}\ \nabla f_{(2)}\ \cdots\ \nabla f_{(n)}]$, $\mathbf{W} = \mathrm{diag}(w_{(1)}\ w_{(2)}\ \cdots\ w_{(n)})$ and $\mathbf{f_t} = [f_{t(1)}\ f_{t(2)}\ \cdots\ f_{t(n)}]^T$ where indices $1...n$ refer to the pixels in neighbourhood Ω.

Finally, we obtain the solution

$$\mathbf{v} = -(\mathbf{GWG}^{\mathrm{T}})^{-1}\mathbf{GWf_t} \quad (5)$$

which is computationally light as $\mathbf{GWG}^T$ is 2×2 and $\mathbf{GWf_t}$ is 2×1.

An image extrapolated with motion vectors obtained with optical flow is shown in Fig. 3. However, this result was not obtained by direct application of (1) but with some further algorithm design, which are discussed next.

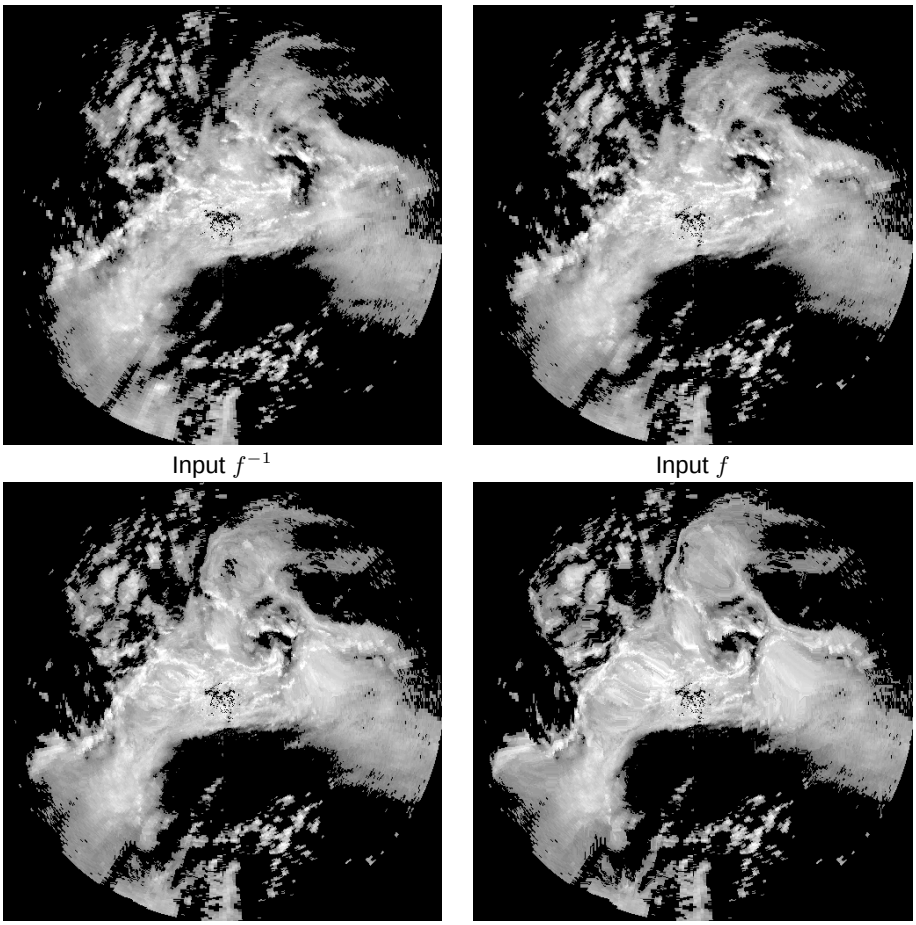


Figure 2: Samples from an image sequence $\{f^{+1}, f^{+2}, ...\}$ extrapolated from f^{-1} and f.

4 Optical flow — implementation

4.1 Preprocessing by smoothing edges

The basic version (1) of optical flow applies well to data that is indeed flow-like: smoothly continuous and hence differentiable. Problems arise when images $f(x, y, t)$ are not continuous but contain distinct objects with crisp edges. Frequently, image object/segment displacements between subsequent image frames may be larger than the scope of the applied neighborhood Ω. Hence, the algorithms tries to match objects against empty image regions and fails to get any clue of directions.

To overcome this problem, one can smoothen the input images with a rectangular averaging window or a gaussian blurring. As a result, the edges will not only become “more differentiable” but also extend the effective scope of the objects.

It should be pointed out that while smoothed images serve as input in motion extration, original unsmoothed data is used in extrapolation. This is illustrated in Fig. 4.1.

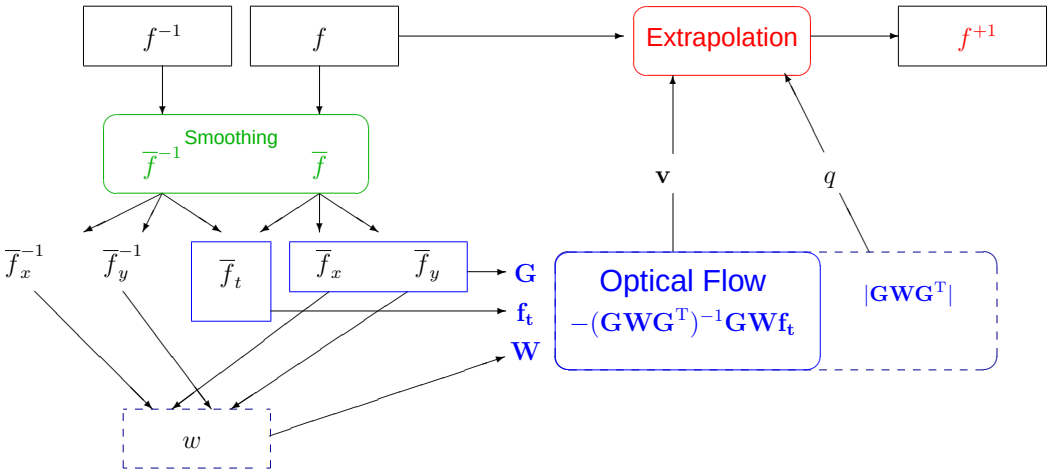


Figure 3: Suggested processing scheme. The previous, current and predicted images are denoted by f^{-1}, f and f^{+1}, respectively.

On the other hand, one should not use too large radii in pre-smoothing because the solution (5) requires that image window Ω contains pixels with independent gradient information (stricly speaking, at least two pixels!).

As as solution, we suggest multicale smoothing, which means blurring the orginal image f with a averaging window operator B and mixing the result in the original image: $\overline{f} = cf + (1-c)B\{f\}$ with some $c \in]0,1[$. (One can apply this recursively.) The resulting image contains both original details and smoothed regions.

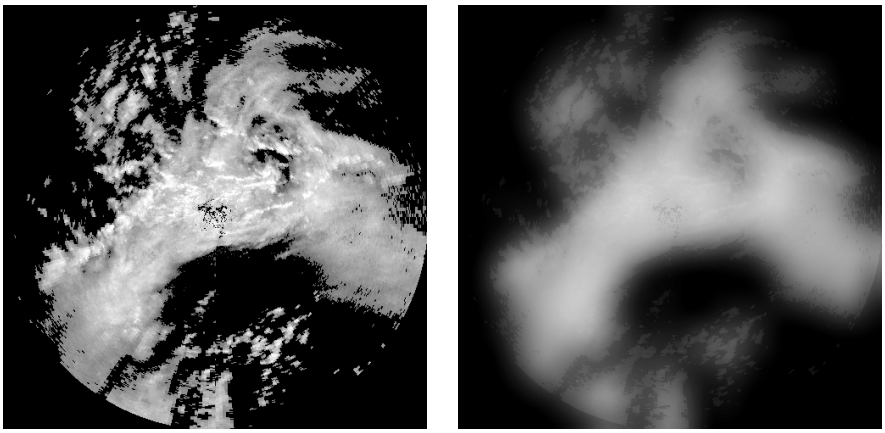


Figure 4: Original and multi-scale smoothed radar image.

.

4.2 Accelerating smoothing by window decomposition

Image processing operations engaging computation of cumulants can be often accelerated. For example, blurring an $N \times N$ image with an $n \times n$ window using a brute force algorithm, hence with four nesting loops (`for i..., for j..., for k..., for l...`), requires a computational effort of order $O(N^2 n^2)$. However, the same result can be obtained by two subsequent averagings of window dimensions $n \times 1$ and $1 \times n$, with effort of $O(N^2 n)$. Moreover, using a *sliding window technique*, that is, by incrementally updating the intensity sum in a window, the computational effort drops down to $O(N^2 + 2n) \approx O(N^2)$ which is remarkable in the case of large n. For example, using 33×33 window, a sliding window technique is about thousand times faster than the brute force implementation!

4.3 Accelerating the computation of $\mathbf{GWG}^{\mathrm{T}}$ and $\mathbf{GWf_t}$

To maximize processing speed, the matrices of optical flow solution **??** should be written out in the program code. Introducting cumulants $G_{xx} = \sum w f_x f_x$, $G_{xy} = \sum w f_y f_x$, $G_{yy} = \sum w f_y f_y$, $G_{xt} = \sum w f_x f_t$, and $G_{yt} = \sum w f_y f_t$ (all within Ω) we can write

$$\mathbf{GWG}^T = \sum_{\Omega} \begin{bmatrix} w f_x f_x & w f_y f_x \\ w f_x f_y & w f_y f_y \end{bmatrix} = \begin{bmatrix} G_{xx} & G_{xy} \\ G_{xy} & G_{yy} \end{bmatrix} \tag{6}$$

and

$$\mathbf{GWf_t} = \sum_{\Omega} \begin{bmatrix} w f_x f_t \\ w f_y f_t \end{bmatrix} = \begin{bmatrix} G_{xt} \\ G_{yt} \end{bmatrix} \tag{7}$$

and finally

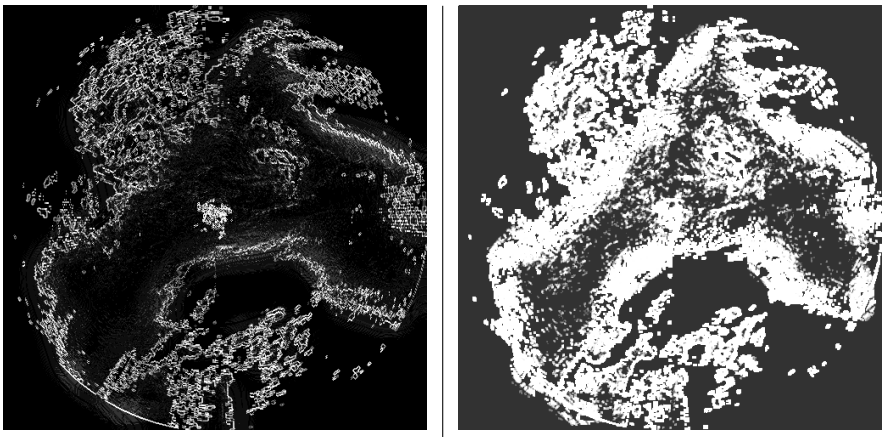
$$\mathbf{v} = \frac{1}{G_{xy}^2 - G_{xx} G_{yy}} \begin{bmatrix} G_{yy} D_x - G_{xy} D_y \\ -G_{xy} D_x + G_{xx} D_y \end{bmatrix} \tag{8}$$

The crucial point here is that if $w = w(x, y, t)$ *in image coordinates* (or a constant), also these cumulants can be updated using a sliding-window technique, yielding complexity $O(N^2 n)$ instead of $O(N^2 n^2)$. (If $w = w(i, j)$ in Ω coordinates, sliding is impossible.) Next, we suggest outlines for $w = w(x, y, t)$.

4.4 Application of quality information

In many applications, we may have quality (confidence, probablity,...) information available. Such information may be provided readily in parallel with input data or the algorithms prosessing the data may generate it with neglible extra effort. The challenge is to pick up good quality functions and well use the information in further data processing.

As quality information for the optical flow algorithm, we suggest using a “gradient stability measure”, for example $w(x, y, t) = ||\nabla f(x, y, t) - \nabla f(x, y, t - 1)||$ (See Fig. 4.1 and Fig. 4.4). Zero or small gradient change tells that that we are (probably) still “sensing the same cloud” while changed gradient means that the cloud has gone past (x, y).



Gradient quality $||\nabla f_2 - \nabla f_1||$ | Velocity quality $|\mathbf{GWG}|$

Figure 5: Examples of quality information that can be extracted in the optical flow computation.

5 Conclusions

In this paper, we discussed motion vector extraction from subsequent images. Although our application context is in weather radars and related forecasting products, the demonstrated techniques are clearly applicable in satellite imageries and in remote sensing imageries more generally.

Autocorrelation is a straightforward, theoretically well-motivated technique for motion detection. At the FMI, it has been succesfully applied in forecasting precipitation up to 2–3 hours. However, since 2D autocorrelation is essentially an image matching technique, it is computationally heavy.

In time critical applications, optical flow should be considered as an alternative method. Also its underlying "physical" model suggest adaptability to meteorological image extrapolation problems.

As with autocorrelation, practical implementation of the core equations of optical flow needs some additional computation to actually work. In this paper, we outlined efficient data smoothing and quality-weighted computation schemes. The results obtained this far encourage continuing this research. In the future, we will develop the overall scheme and study parameter selection. As to applications, we will study separate processing for convective and widespread rain as well as detection of divergence and convergence.

References

Holmlund, K. (2000). The atmospheric motion vector retrieval scheme for meteosat second generation, *Proceedings of 5th International Winds Workshop*, Eumetsat.

J. L. Barron, S. S. B. and Fleet, D. J. (1994). On optical flow, *in* I. Plander (ed.), *6th Int. Conf. on Artificial Intelligence and Information-Control Systems of Robots (AIICSR)*, World Scientific, Bratislava, Slovakia, pp. 3–14. Sept. 12-16, 1994, Smolenice Castle, Slovakia.

Sonka, M., Hlavac, V. and Boyle, R. (1993). *Image Processing, Analysis and Computer Vision*, Chapman & Hall Computing.