Tutorial of Motion Estimation Based on Horn-Schunk Optical Flow Algorithm in MATLAB®

Darun Kesrarat[1] and Vorapoj Patanavijit[2]

[1] Department of Information Technology, Faculty of Science and Technology
[2] Department of Computer and Network Engineering, Faculty of Engineering
Assumption University, Bangkok, Thailand
E-mail: <darun@scitech.au.edu; patanavijit@yahoo.com>

Abstract

The Horn-Schunk algorithm (HS) is one of the classical algorithms in optical flow due to its reasonable performance and simplicity of the algorithm. This article presents in detail the process of the HS algorithm and its step by step coding in MATLAB®. The HS algorithm is a technique used to identify the image velocity or motion vector based on Spatial Temporal Gradient Technique which computes the image velocity from spatiotemporal derivatives of image intensity. Firstly, the original estimated intensity for gradient constraint on the image sequence is obtained by using the gradient constraint with a global smoothness. Then, iterative equations are solved to minimize the sum of the errors for the rate of change of image gradient intensity and obtain the image velocity.

Keywords: *Spatial Temporal Gradient Technique, gradient intensity, motion vector, image velocity.*

Introduction

Optical flow is a technique used for many particular fields, such as motion estimation to predict the motion vector of a moving object, video compression and reconstruction for reducing temporal redundancy present in frame sequences and allowing a better compression of video material, and image segmentation for tracking a moving object. In conventional predictive methods for motion estimation, the difference between the current frame and the predicted frame, based on a previous frame (motion vector, or MV), is coded and transmitted; then it is used to reconstruct a higher resolution still image or video sequence from a sequence of low resolution images in achieving super-resolution (Kesrarat and Patanavijit 2011). This article concentrates on a classical optical flow algorithm, namely the Horn-Schunck algorithm (HS) proposed by Horn and Schunck (1981). The HS algorithms are the most popular differential algorithms which have been applied for many applications and have been referenced for many performance evaluation models. Barron, Fleet and Beauchemin (BFB) adjusted the kernel model for performance evaluation over HS algorithms, but its focus is on the density of velocity (Barron *et al*. 1994). This article explains the motion estimation algorithm based on the Horn-Schunk optical flow algorithm (HS), which applies the kernel of BFB, and also represents the algorithm step by step in MATLAB®.

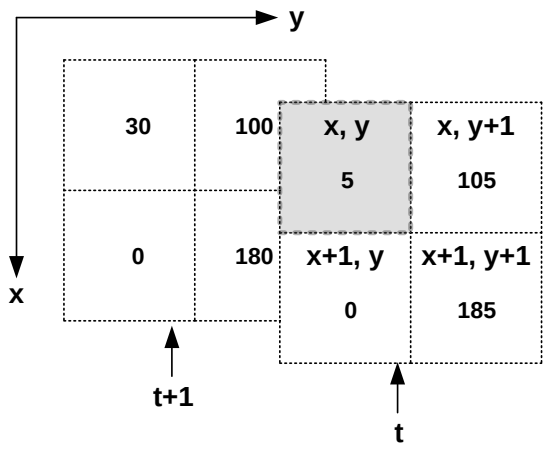
Horn-Schunk Algorithm (HS)

This algorithm is based on a differential technique computed by using a gradient constraint (brightness constancy) with a global smoothness to obtain an estimated velocity field (Horn and Schunk 1981). There are two main processes for the implementation of the HS algorithm. The first one is an estimation of partial derivatives, and the second one is a minimization of the sum of the errors by an iterative process to present the final motion vector.

Estimation of classical partial derivatives

This section presents the estimation process of the classical derivatives of image intensity or brightness from the image sequence. The brightness of each pixel is constant along its motion trajectory in the image sequence. The relationship in continuous images sequence will be taken into account to estimate the original intensity for a gradient constraint. Let $I(x,y,t)$ denote the gradient intensity (brightness) of point (x,y) in the images at time t. In each image sequence, I_x, I_y, and I_t are computed for each pixel (see also Figs. 1-2):

$$I_x = ¼ \{I_{x,y+1,t} - I_{x,y,t} + I_{x+1,y+1,t} - I_{x+1,y,t} + I_{x,y+1,t+1} - I_{x,y,t+1} + I_{x+1,y+1,t+1} - I_{x+1,y,t+1}\},$$
$$I_y = ¼\{I_{x+1,y,t} - I_{x,y,t} + I_{x+1,y+1,t} - I_{x,y+1,t} + I_{x+1,y,t+1} - I_{x,y,t+1} + I_{x+1,y+1,t+1} - I_{x,y+1,t+1}\},$$
$$I_t = ¼ \{I_{x,y,t+1} - I_{x,y,t} + I_{x+1,y,t+1} - I_{x+1,y,t} + I_{x,y+1,t+1} - I_{x,y+1,t} + I_{x+1,y+1,t+1} - I_{x+1,y+1,t}\}. \quad (1)$$



I_x = ((105 – 5) + (185 – 0) + (100 –3 0) + (180 – 0)) / 4 = 133.75
I_y = ((0 – 5) + (185 – 105) + (0 – 30) + (180 – 100)) / 4 = 31.25
I_t = ((30 – 5) + (0 – 0) + (100 – 105) + (180 – 185)) / 4 = 3.75

Fig. 1. The partial derivatives of image brightness at point (x, y).

Estimation of partial derivatives on BFB kernel

Barron *et al*. (1994) proposed performance evaluation over many algorithms of optical flow and modification of some of the variant variables. We focus on the kernel of mask coefficient for gradient estimation which is the core functional of the HS algorithms. The gradient estimation kernel of the BFB model uses 4-point central differences for differentiation defined as (see also Figs. 3-7):

$$I_x = 1/12 \{-1 \times I_{x,y-2} + 8 \times I_{x,y-1} + 0 \times I_{x,y} - 8 \times I_{x,y+1} + 1 \times I_{x,y+2}\},$$
$$I_y = 1/12 \{-1 \times I_{x-2,y} + 8 \times I_{x-1,y} + 0 \times I_{x,y} - 8 \times I_{x+1,y} + 1 \times I_{x+2,y}\},$$
$$I_t = 1/12 \{-1 \times I_{x,y,t-2} + 8 \times I_{x,y,t-1} + 0 \times I_{x,y,t} - 8 \times I_{x,y,t+1} + 1 \times I_{x,y,t+2}\}. \quad (2)$$

It is known that in most situations of motion estimation one may use only two frames to calculate I_t as shown in Fig. 6. As a result of using the BFB kernel, it presents more stability of original gradient estimation than the result of performance evaluation in Barron *et al*. (1994).

```
% im1 is gray image sequence at time t
% im2 is gray image sequence at t+1

Ix = conv2(im1,0.25* [-1 1; -1
1],'same') + conv2(im2, 0.25*[-1 1; -1
1],'same');

Iy = conv2(im1, 0.25*[-1 -1; 1 1],
'same') + conv2(im2, 0.25*[-1 -1; 1
1], 'same');

It = conv2(im1, 0.25*ones(2),'same') +
conv2(im2, -0.25*ones(2),'same');
```

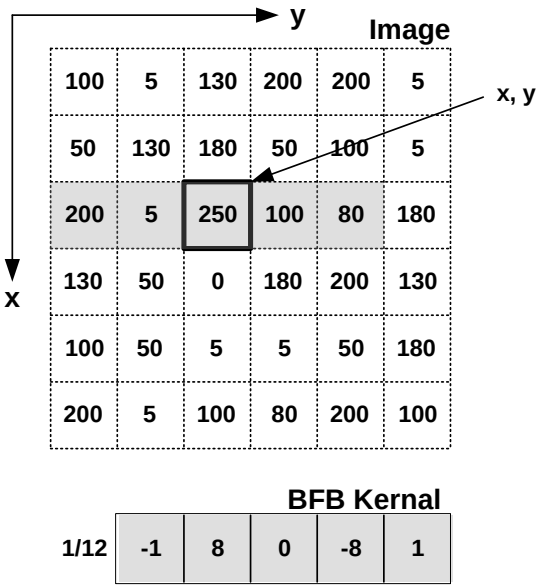
Fig. 2. MATLAB® code for gradient estimation with original partial derivatives.

1/12	-1	8	0	-8	1

Fig. 3. The kernel coefficient of BFB.

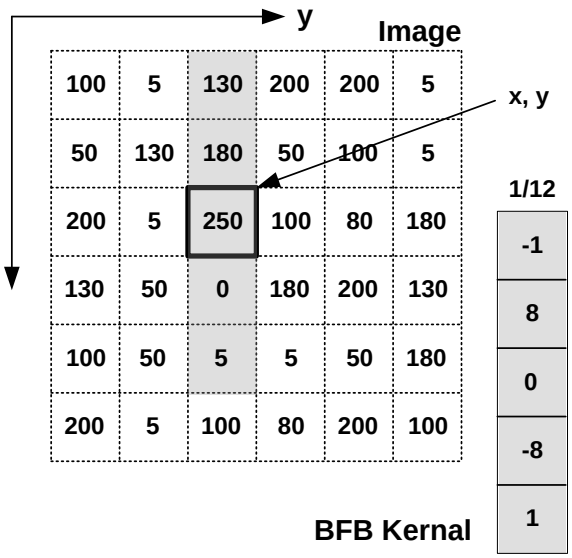
```
% im1 is gray image sequence at time t
% im2 is gray image sequence at t+1

Ix= conv2(im1,(1/12)*[-1 8 0 -8 1],
 'same');

Iy= conv2(im1,(1/12)*[-1 8 0 -8 1]',
 'same');

It = conv2(im1, ones(1),'same') +
 conv2(im2, -ones(1),'same');

% modify ft because there are only 2 frames
which are used for calculation
```

Fig. 4. MATLAB® code for gradient estimation with BFB kernel of 2-image sequence.



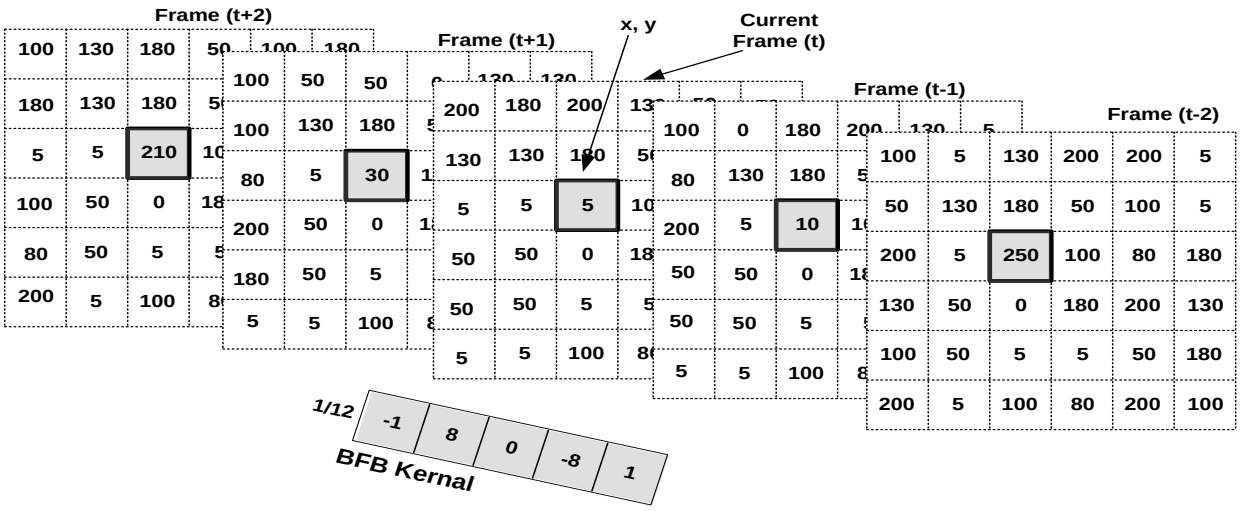
$I_x(2,2)$ = (-1 x 200 + 8 x 5 + 0 x 250 – 8 x 100 + 1 x 80) / 12 = -73.33

Fig. 5. An illustration of gradient calculation for I_x of BFB kernel coefficient.



$I_y(2,2)$ = (-1 x 130 + 8 x 180 + 0 x 250 – 8 x 0 + 1 x 5) / 12 = 109.58

Fig. 6. An illustration of gradient calculation for I_y of BFB kernel coefficient.



$I_t(2,2)$ = (-1 x 210 + 8 x 30 + 0 x 5 – 8 x 10 + 1 x 250) / 12 = 16.66

Fig. 7. An illustration of gradient calculation for I_t of BFB kernel coefficient.

Minimization

In practice, the image intensity or brightness measurement may be corrupted by quantization or noise. According to the equation for the rate of change of image brightness:

$$I(x,y,t) = I(x+u, y+v, t+1), \quad (3)$$

$$\varepsilon = u\, I_x + v\, I_y + I_t = 0, \quad (4)$$

where u and v are the horizontal and vertical motion vectors of optical flow, respectively, one cannot expect ε to be zero. The problem is to minimize the sum of errors in the equation for the rate of change of image brightness as near as 0. So, the smoothness weight (α) is iteratively presented as (see also Fig. 8):

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

$$v^{k+1} = \bar{v}^k - \frac{I_y\left[I_x\bar{u}^k + I_y\bar{v}^k + I_t\right]}{\alpha^2 + I_x^2 + I_y^2}, \quad (5)$$

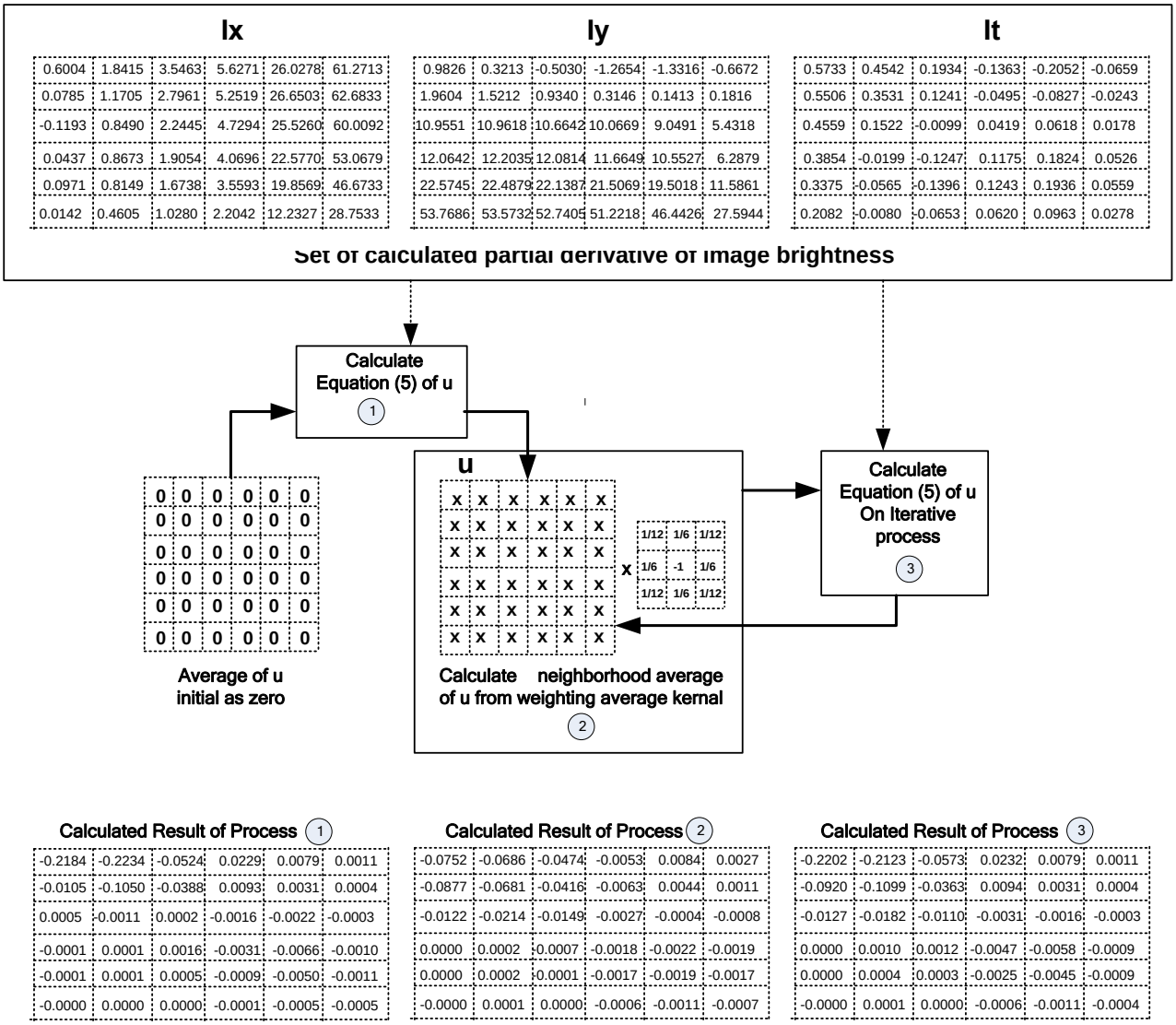


Fig. 8. An illustration of the iterative minimization process for u where steps 2 and 3 are iterative.

where $\bar{u}^k$ and $\bar{v}^k$ denote horizontal and vertical neighborhood averages (u^k and v^k), which initially are set to zero and then the weighted average of the value at neighboring points based on the kernel in Fig. 8 is applied for further iterations using Eq. (5) as illustrated in Figs. 9-10. The smoothness weight (α) plays an important role where the brightness gradient is small, for which the suitable value should be determined.

1/12	**1/6**	**1/12**
1/6	**-1**	**1/6**
1/12	**1/6**	**1/12**

Fig. 9. Weighted average kernel at neighboring points.

```
% Set initial value of u and v to zero
u = 0;
v = 0;

% Weighted Average kernel
kernel=[1/12 1/6 1/12;1/6 0 1/6;1/12
 1/6 1/12];

%Minimizing Iterations (100 times)
for i=1:100

%Compute local averages of the vectors

 uAvg=conv2(u,kernel,'same');
 vAvg=conv2(v,kernel,'same');

%Compute flow vectors constrained by the local
averages and the optical flow constraints,
where alpha is the smoothing weight

 u = uAvg - ( Ix .* ( ( Ix .* uAvg ) +
( Iy .* vAvg ) + It ) ) ./ ( alpha^2 +
Ix.^2 + Iy.^2);

 v = vAvg - ( Iy .* ( ( Ix .* uAvg ) +
( Iy .* vAvg ) + It ) ) ./ ( alpha^2 +
Ix.^2 + Iy.^2);

end
```

Fig. 10. MATLAB® code of the iterative minimization process.

Conclusion

According to the characteristic of the HS algorithm, when applied with the BFB kernel it provides simplicity in the algorithm with reasonable performance and better quality, but the value of the smoothing weight (α) cannot be defined exactly because the suitable value is varying upon different image sequences. As a consequence, the suitable iteration times also cannot be defined for the best outcome, which impacts the processing time for the best motion vector at the output. Figures 11-14 show the results for the motion vector of the HS algorithm from the partial representation of Foreman, Coastgurad, Akiyo and Container video sequences, respectively. The results of the motion vector identify the direction of pixel movement from the two different consecutive frames under the smoothing weight α = 10 and 100 iterations of the minimizing process.

References

Barron, J.L.; Fleet, D.J.; and Beauchemin, S.S. 1994. Performance of optical flow techniques. International Journal of Computer Vision 12(1): 43-77.

Horn, B.K.P.; and Schunck, B.G. 1981. Determining optical flow. Artificial Intelligence 17(1-3): 185-203.

Kesrarat, D.; and Patanavijit, V. 2011. Performance analysis on weighting factor (α) on spatial temporal gradient technique and high confidence reliability with sub-pixel displacement. Proc. 8th International Conference on Electrical Engineering/ Electronics, Computer, Telecommunications and Information Technology (ECTI-CON 2011), Khon Kaen, Thailand, 17-19 May 2011, pp. 1,043-46.

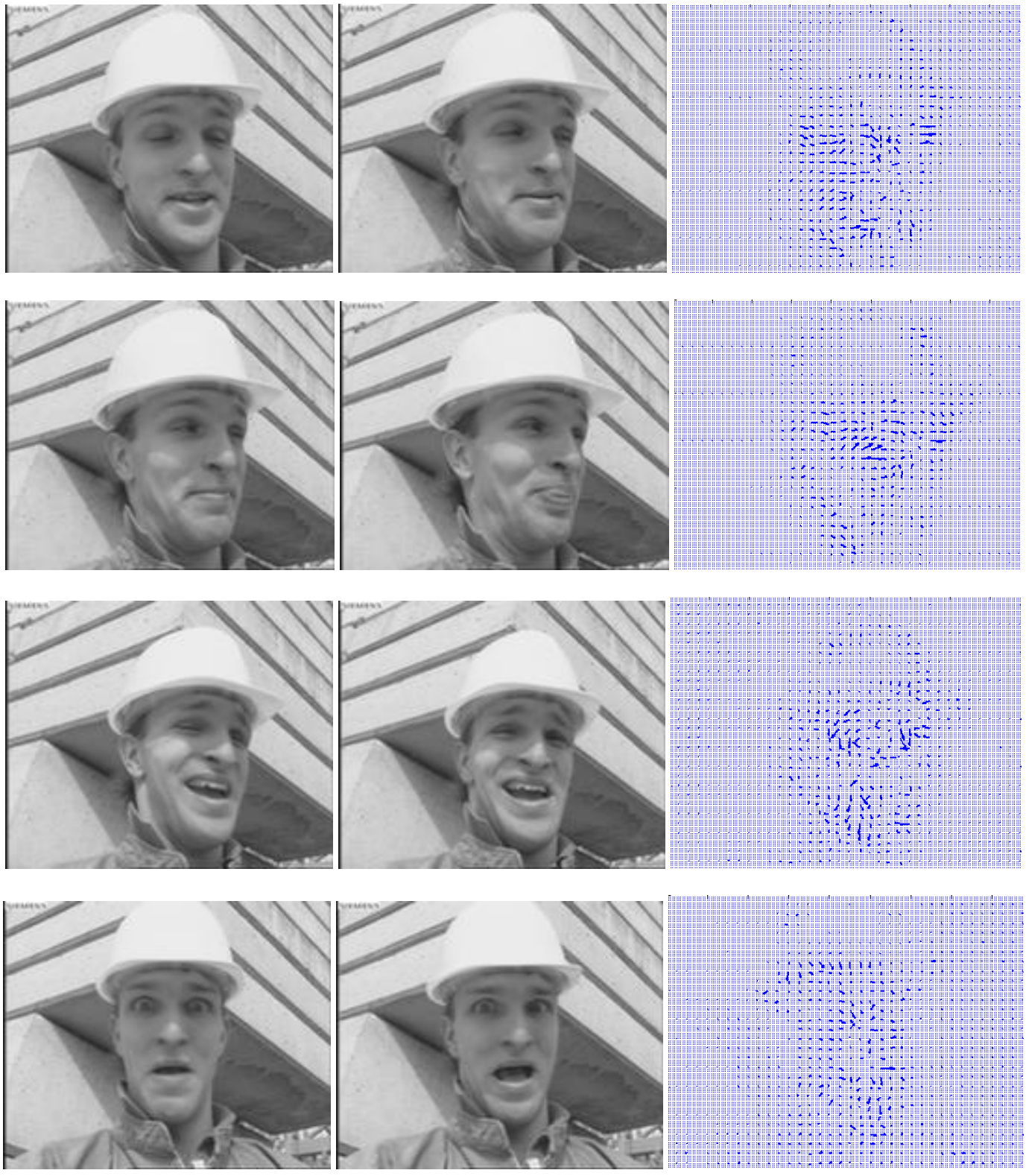


Fig. 11. Examples of the motion vector of the HS algorithm from a part of FOREMAN sequence.

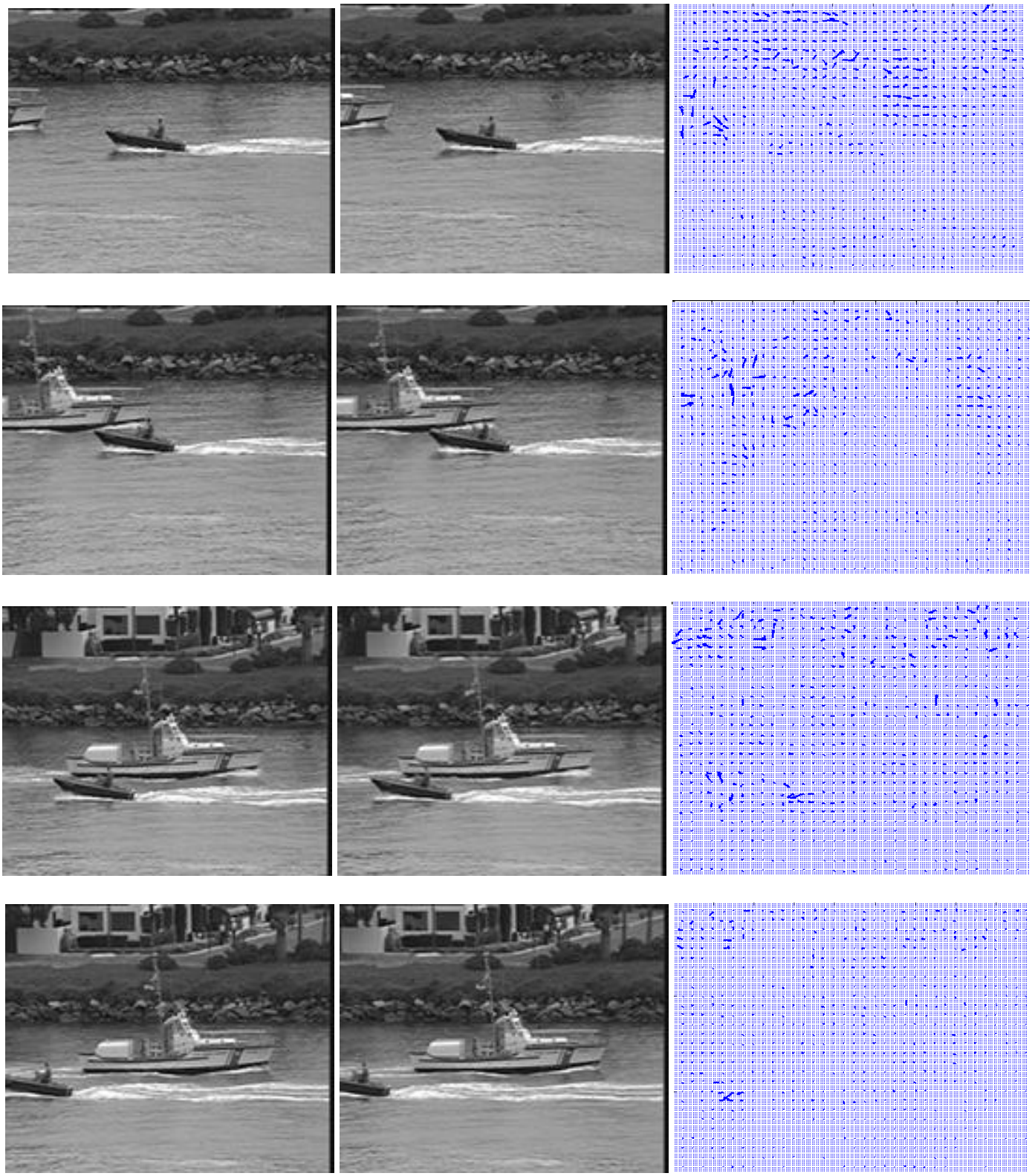


Fig. 12. Examples of the motion vector of the HS algorithm from a part of COASTGUARD sequence.

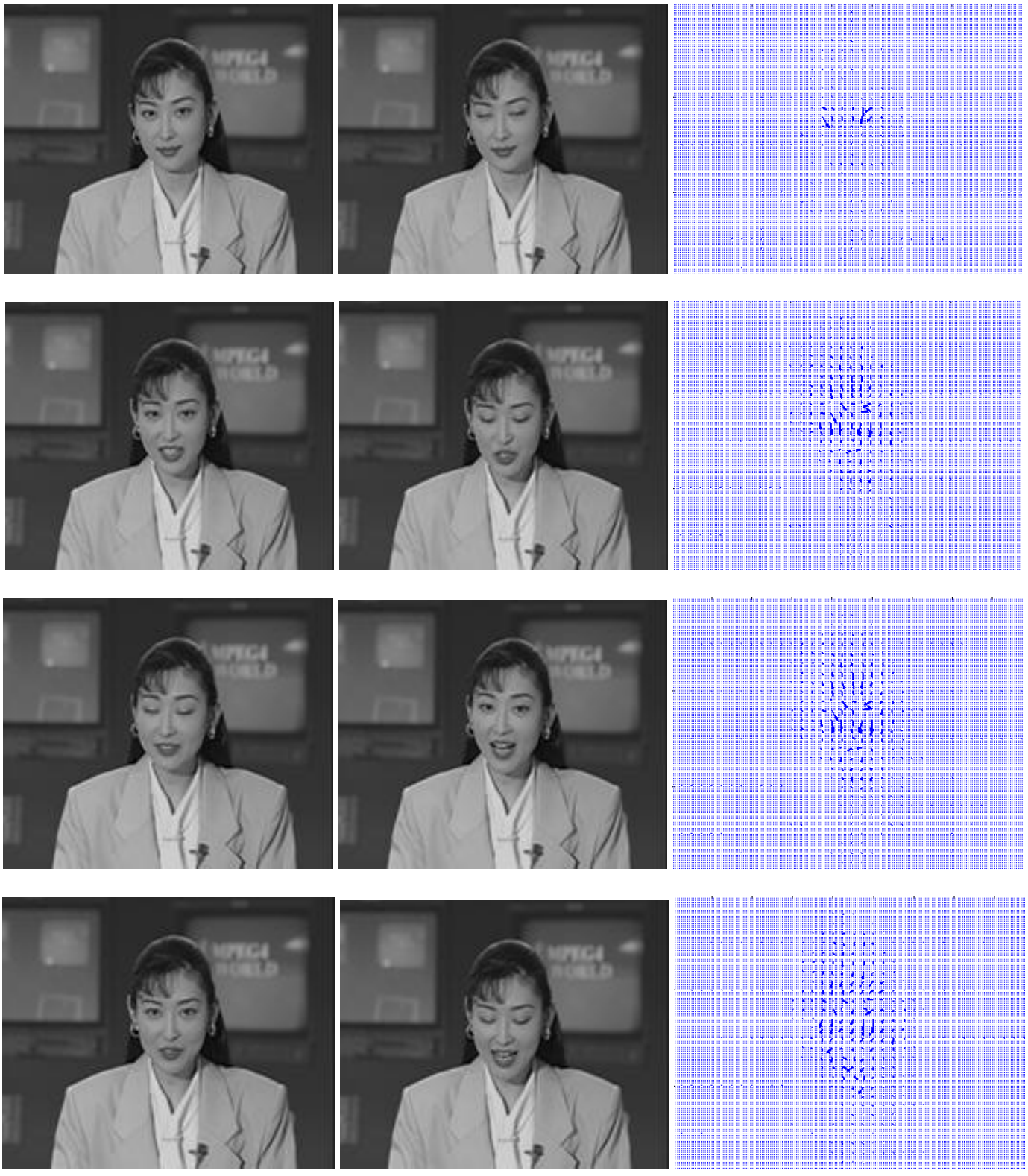


Fig. 13. Examples of the motion vector of the HS algorithm from a part of AKIYO sequence.

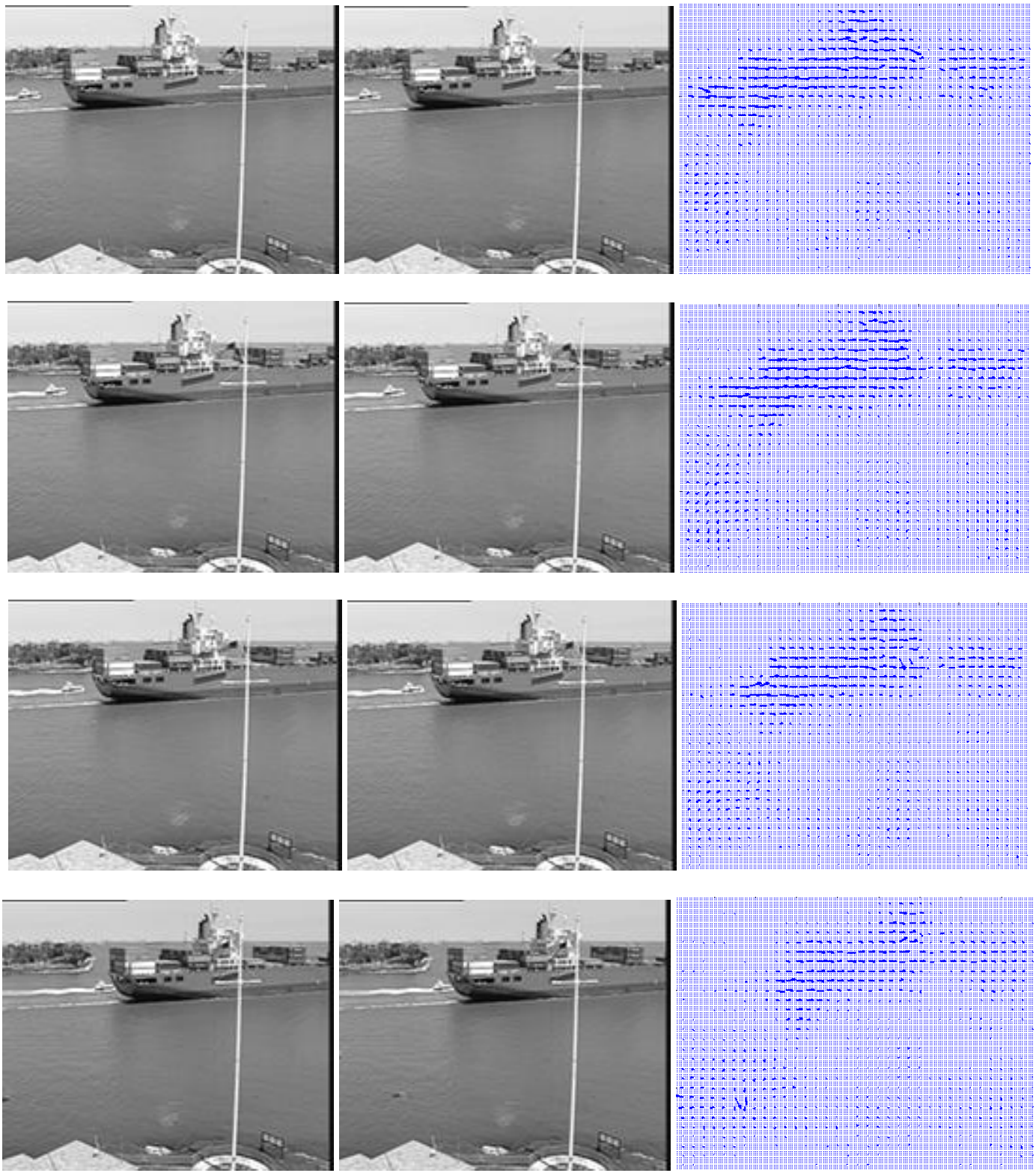


Fig. 14. Examples of the motion vector of the HS algorithm from a part of CONTAINER sequence.