A Motion/Disparity Vector Field Smoothing Algorithm

H. Sahinoglou, S. Malassiotis, N. Nikolaidis, N. Grammalidis
M. G. Strintzis, I. Pitas
Department of Electrical Engineering
University of Thessaloniki
54006 Thessaloniki, Greece

ABSTRACT

Disparity and motion vector fields obtained by block matching methods, contain noisy vectors, especially in uniform background regions where matching is ambiguous and motion boundaries. This leads to a considerable increase in bit rate which can be avoided by smoothing the vector field after it has been estimated. Also by incorporating a smoothing function in the block matching process, we can get efficient algorithms that require no postprocessing steps.

A smoothing method was considered, where we try to improve the vectors produced by another method using the following three criteria:

1) Small prediction error.
2) Smooth vector field (small differences between vectors and their neighbours).
3) Efficient vector encoding by favoring small vectors that are encoded efficiently using Huffman tables.

To achieve all of these goals, we set out to minimize the following objective function for each vector:

\[ E = \sum_{i=1}^{K} |x_k - d_k(x_k)| + \alpha \sum_{i=1}^{8} (|v_i.x - v.x_k| + |v_i.y - v.y_k|) + \beta (|v.x| + |v.y|) \]

where we have K (=64) pixels in the block, \( x_k \) is the kth pixel, \( d_k(x_k) \) is the prediction of the previous frame based on vector \( v = [v.x, v.y]^T \), and \( v_1...v_8 \) are the eight neighbouring vectors. That makes each of the factors of this equation favor one of each consideration mentioned above, with relative weights \( \alpha \) and \( \beta \) for the second and third, relative to the first. By the adjusting parameters \( \alpha, \beta \) we can compensate between good smoothing and small error.

The problem with this function is that we cannot define derivatives of this function relative to \( v.x \) and \( v.y \) to generate a standard gradient iterative method. While this is possible for the second and third terms if we use the \( L_2 \) norm, it is impossible for the first term even then (except for sub-pixel ranges, which do not give room for much improvement).

Therefore the \( L_1 \) norm is retained to allow a faster implementation, and instead we use an approach similar to the one discussed in [Sti]. In each iteration we use a list of candidates to replace the given vector, and choose the one minimizing the function. The candidates are:
1) The vector itself.

2) Small perturbations ±1/2 pixel in the x or y coordinate giving 4 candidates.

3) The eight neighboring elements of the vector (8 candidates).

4) The average of the vector and its neighbors.

This leads to 14 function evaluations per vector per iteration (no revaluation is done for case 1). The implementation of this method took about 4.5 seconds/iteration, and was tried in various block matching methods as a post-processing step. The following block matching algorithms were used to create motion/disparity vector fields:

- Full search.
- 3-Step search.
- Logarithmic search.
- A multiresolution algorithm [Tso].
- A recursive least squares correlation algorithm [Gru].

Results obtained with real images, show that the vector field components entropy can be reduced up to 20% for motion and 10% for disparity without serious degradation of the reconstructed images.

The same method was also used in conjunction with the requirement that for two sequential stereo image pairs in times t and t+1 the following equation holds for the same 3-D point projected on the four images:

$$D_{t+1} - D_t = M_t - M_t$$

where $D$ stands for disparity and $M$ for motion. The deviation from equality was used as an extra term in the error function $E$. The addition of this constraint led to improvement in comparison with the previous method.

Finally the smoothing problem was also solved by minimizing the following error function with respect to $V$:

$$e(x,y;V) = \sum_{i=0}^{M} \sum_{j=0}^{N} \left( I_1[yM + i, xN + j] - I_2[yM + i + V_x, xN + j + V_y] \right)^2$$

$$+ a \sum_{i=-1}^{1} \sum_{j=-1}^{1} \| V - U[y + i, x + j] \|_2^2 + b\| V \|_2^2$$

A steepest descent algorithm was used to perform the optimization.

References


[Tso] D. Tsavaras, M.G. Strintzis, H. Sabiroglou "A Multiresolution Technique for Block Matching in Motion and Disparity Estimation" (Submitted for publication)