Template matching based on the L_p norm using sufficient conditions with incremental approximations

Federico Tombari Stefano Mattoccia
Luigi Di Stefano
Department of Electronics Computer Science and Systems (DEIS)
Viale Risorgimento 2, 40136 - Bologna, Italy
Advanced Research Center on Electronic Systems (ARCES)
Via Toffano 2/2, 40135 - Bologna, Italy
University of Bologna
{ ftombari, smattoccia, ldistefano}@deis.unibo.it

Abstract

This paper proposes a novel algorithm aimed at speeding-up template matching based on the L_p norm. The algorithm is exhaustive, i.e. it yields the same results as a Full Search *(FS) template matching process, and is based on the deployment of tight lower bounds that can be derived by using together the triangular inequality and partial evaluations of the L_p norm. In order to deploy this,* template *and* image subwindows *are properly partitioned. The experimental results prove that the proposed algorithm allows speeding-up the FS process and also (when applied to the L_2 norm) the exhaustive approach based on the Fast Fourier Transform.*

1. Introduction

Locating a given template into an image is the task of template matching, a basic image analysis technique deployed in a large variety of machine vision applications. The *Full Search* (FS) template matching process relies on calculating at each position of the image a function measuring the degree of similarity/dissimilarity between the template and the portion of the image currently under examination, which will be referred to in the following as *image subwindow* (see also [10]). Within this scope, widely used dissimilarity functions are the *Sum of Squared Distances* (SSD) and the *Sum of Absolute Differences* (SAD).

Since the *Full Search* (FS) approach is known to be computationally very expensive, several algorithms have been proposed in the literature in order to speed-up the matching process. These can be divided into exhaustive and non-exhaustive algorithms. The latter aim at speeding-up the matching process by reducing the overall search space, thus not guaranteeing the final result to be the same as with the FS approach (i.e. SSDA [1], [8]). On the other hand, exhaustive algorithms are able to yield exactly the same result as the FS process. Within these algorithms, FFT-based methods carry out part of the computations required by the SSD function in the frequency domain, while [3] proposed a signal domain method based on pruning rapidly unsatisfactory matching positions for both SAD and SSD-based template matching processes. Recently an original approach based on projection kernels and the Walsh-Hadamard transform has been proposed in [4].

Furthermore, template matching can be regarded as a Nearest-Neighbour Search problem, which associates this topic with other relevant research fields such as Block Matching for motion estimation and Vector Quantization, where several exhaustive and non-exhaustive techniques have been proposed in the last two decades. For instance, in the motion estimation field [6] and [9] proposed signal domain methods based on the triangular inequality, while [5] and [2] improved these methods by proposing a multiresolution framework similar to the one used by [3] in its technique for template matching.

In this paper we propose a novel signal domain algorithm that meets the exhaustivity requirement and is based on the deployment of several sufficient conditions -characterized by increasing efficiency- for pruning rapidly unsatisfactory matching positions. The algorithm relies on partitioning the template and the image subwindow so as to determine tight lower bounds of the L_p norm by means of applications of the triangular inequality together with partial evaluations of the L_p norm itself.

Proceedings of the IEEE International Conference
on Video and Signal Based Surveillance (AVSS'06)
0-7695-2688-8/06 $20.00 © 2006 IEEE

2. Proposed algorithm

Let the template, T, be of size $M \times N$, and the image, $I(x, y)$, of size $W \times H$, where (x, y) denotes the generic image coordinates. We also indicate the image subwindow located at (x, y) as $I_s(x, y)$. Then, the generic *distance* function measuring the dissimilarity between the template and the image subwindow can be written as follows:

$$\delta_p(x,y) = ||I_s(x,y) - T||_p^p = \\ = \sum_{i=1}^{M}\sum_{j=1}^{N} \left|I(x+i,y+j) - T(i,j)\right|^p \quad (1)$$

with $||\cdot||_p$ denoting the L_p norm. If $p = 1$ then $\delta_p(x, y)$ coincides with the SAD function, while by taking $p = 2$ with the SSD function.

Let's now consider the *triangular inequality*:

$$||I_s(x,y) - T||_p \geq \left|||I_s(x,y)||_p - ||T||_p\right| \quad (2)$$

that allows the definition of a function, $\beta_p(x, y)$, which turns out to be a lower-bound of $\delta(x, y)$ at any (x, y):

$$\beta_p(x,y) = \left|||I_s(x,y)||_p - ||T||_p\right|^p \quad (3)$$

We propose to partition the template and the image subwindow into r disjoint regions. In order to simplify the implementation of the method, we use regions made out of successive rows (see Fig. 1). Furthermore, to increase the computational efficiency of the algorithm (see Section 3), we use regions characterized by the same number of rows, n, except for the last one. In the particular case that N is a multiple of r, the regions will have all the same size. Based on the described partitioning scheme, we define the generic *partial bound* term computed between rows (ρ, θ) as:

$$\beta_p(x,y)|_\rho^\theta = \left|||I_s(x,y)||_p\right|_\rho^\theta - ||T||_p\Big|_\rho^\theta\Big|^p = \\ \left|\left[\sum_{i=1}^{M}\sum_{j=\rho}^{\theta}\left|I(x+i,y+j)\right|^p\right]^{\frac{1}{p}} - \left[\sum_{i=1}^{M}\sum_{j=\rho}^{\theta}\left|T(i,j)\right|^p\right]^{\frac{1}{p}}\right|^p \quad (4)$$

and the generic *partial distance* term between rows (ρ, θ) as:

$$\delta_p(x,y)|_\rho^\theta = ||I_s(x,y) - T||_p^p\Big|_\rho^\theta = \\ = \sum_{i=1}^{M}\sum_{j=\rho}^{\theta}\left|I(x+i,y+j) - T(i,j)\right|^p \quad (5)$$

where, of course, the following inequality applies:

$$\delta_p(x,y)|_\rho^\theta \geq \beta_p(x,y)|_\rho^\theta \quad (6)$$

By calculating (4) on each region defined on the template and image subwindow by the described partitioning scheme

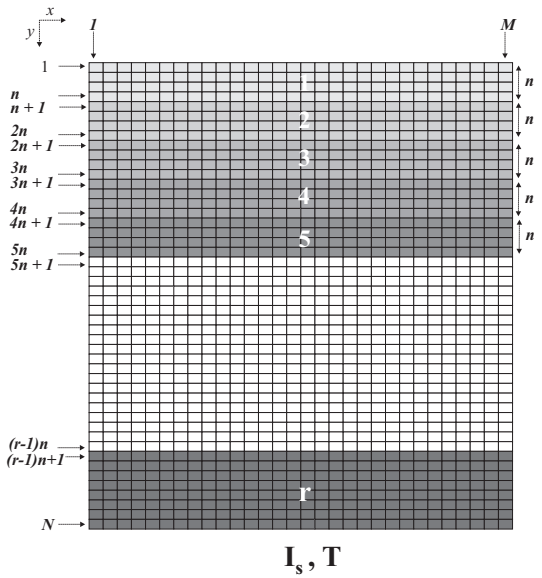

Figure 1. Partitioning scheme of I_s and T.

we obtain r different partial bound terms. Then, summing up these terms yields:

$$\beta_{p,r}(x,y,n)|_1^N = \sum_{t=1}^{r-1}\left[\beta_p(x,y)|_{(t-1)\cdot n+1}^{t\cdot n}\right] + \beta_p(x,y)|_{(r-1)\cdot n+1}^{N} \quad (7)$$

Instead, by calculating (5) on each region and summing up we get:

$$\delta_p(x,y) = \sum_{t=1}^{r-1}\left[\delta_p(x,y)|_{(t-1)\cdot n+1}^{t\cdot n}\right] + \delta_p(x,y)|_{(r-1)\cdot n+1}^{N} \quad (8)$$

Thus, the $\beta_{p,r}(x, y, n)|_1^N$ function represents a lower bound of the $\delta_p(x, y)$ function.

The bounding property of function $\beta_{p,r}(x, y, n)|_1^N$ can be deployed within a template matching process in order to prune rapidly unsatisfactory matching positions. In particular, a sufficient condition for pruning the current image position is given by:

$$\beta_{p,r}(x,y,n)|_1^N > \delta_m \quad (9)$$

where δ_m is the minimum of the distance function *found so far*. In fact, if inequality (9) holds, the current image position cannot improve the degree of similarity to the template with respect to the current best matching position; thus, the current image position can be pruned without computing the actual distance function $\delta_p(x, y)$.

In case (9) is not verified, the algorithm determines a more efficient sufficient condition characterized by a

Proceedings of the IEEE International Conference
on Video and Signal Based Surveillance (AVSS'06)
0-7695-2688-8/06 $20.00 © 2006 IEEE

bounding function closer to the actual value of the $\delta_p(x,y)$ function. This is done by computing the partial distance term associated with the first region and then using it to replace the corresponding partial bound term in $\beta_{p,r}(x,y)$:

$$\gamma_{p,(r-1)}(x,y,n) = \delta_p(x,y)|_1^n + \beta_{p,(r-1)}(x,y,n)|_{n+1}^N \tag{10}$$

As a result, the new sufficient condition that can be checked in order to prune the current position is given by:

$$\gamma_{p,(r-1)}(x,y,n) > \delta_m \tag{11}$$

The better efficiency of condition (11) is guaranteed by the following inequalities:

$$\delta_p(x,y) \geq \gamma_{p,(r-1)}(x,y,n) \geq \beta_{p,r}(x,y,n)|_1^N \tag{12}$$

that can be easily inferred from the properties of partial bound and partial distance terms.

Should (11) be not verified either, the algorithm would continue checking other sufficient conditions characterized by increasing efficiency by substituting, at generic step i, the $i-th$ partial bound term with its corresponding partial distance term. Proceeding this way, the algorithm can determine and check up to r sufficient conditions. The last lower bounding function that can be determined is:

$$\gamma_{p,1}(x,y,n) = \delta_p(x,y)\big|_1^{(r-1)\cdot n} + \beta_{p,1}(x,y,n)\big|_{(r-1)\cdot n+1}^N \tag{13}$$

and the associated sufficient condition:

$$\gamma_{p,1}(x,y,n) > \delta_m \tag{14}$$

Should (14) be not verified, the algorithm would complete the computation of the distance function by calculating the partial distance term associated with the last region and then compare $\delta_p(x,y)$ to δ_m.

3. Considerations

The proposed technique has been described here for typical template matching scenarios, where the best match together with its score has to be found within an image. Nevertheless, it is worth noting that the application to other fields where a threshold δ_t is fixed previously to the matching process in order to select matching candidates, such as *image filtering* [4], is straightforward. In fact this can be easily done by initializing δ_m as δ_t and then storing into an array all the positions (x,y) that, not being skipped by any of the sufficient conditions applied, verify:

$$\delta_p(x,y) < \delta_m \tag{15}$$

To briefly compare our technique with other proposals, it is worth noting that [3] uses sufficient conditions based on the triangular inequality at different levels of resolution. It does not incorporate partial distance terms into the bounding functions, this implying that when no sufficient condition is verified the overall distance function has to be calculated from scratch. Algorithms [6] and [9] use $\beta_p(x,y)$ as a way to determine a sufficient condition for pruning non-matching positions. Since with $p = 1, 2$ holds the following inequality (see Appendix A):

$$\beta_p(x,y) \leq \beta_{p,r}(x,y,n)|_1^N \tag{16}$$

the lower bounds proposed in this paper represent a closer approximation of the $\delta_p(x,y)$ term. Moreover, it is worth noting that the partial bound terms, which are the basic terms of our bounding functions, can be efficiently computed thanks to standard incremental techniques such as *box-filtering* [7]. Generally speaking, the number of distinct box-filters required by the algorithm is equal to the number of different sized regions used within the partitioning scheme. As discussed in Section 2, with only two different sized regions only two distinct box-filters are required by the algorithm, thus reducing the computational overhead as well as the memory footprint. When the template and image subwindow can be partitioned into regions having all the same size ($n = \frac{N}{r}$), the computational efficiency increases furtherly since only one box-filter is needed. It is worth pointing out that any different partitioning scheme which subdivides the image window and the template into equally sized blocks could be applied, and would lead to the same cost in terms of incremental calculations. Obviously, increasing the number of subsets improves the tightness of the bounding function though requires a higher number of operations for its calculation. Hence, this turns out to be a trade-off for the choice of the granularity of the partitioning scheme.

The proposed algorithm, as it is the case of [3], [2], [4], [5], [6] and [9], is data dependent. Since the number of pruned positions can not be foreseen, the amount of computation required by our algorithm cannot be estimated in advance. In particular, the execution time depends significantly on the position of the best matching image subwindow within the search area as well as on a proper choice of the two parameters (r,n) associated with the partitioning scheme. In order to alleviate these dependencies, we have developed two methods aimed at estimating respectively the position of the best matching image subwindow and the optimal choice of parameters (r,n). These methods,

Proceedings of the IEEE International Conference
on Video and Signal Based Surveillance (AVSS'06)
0-7695-2688-8/06 $20.00 © 2006 IEEE

not described here for the sake of brevity, require a negligible computational cost and hence can be used at run-time to provide initialization values to our template matching algorithm.

Table 1. Speed-ups yielded by our algorithm vs. FS algorithm, SSD case.

Dataset	$(r, \frac{n}{N})$	$Speed-up$	$P_{tot}[\%]$	$P_0[\%]$
paint1	(20,0.05)	108.7	≈ 100.0	91.3
paint2	(16,0.06)	136.7	≈ 100.0	96.9
paint3	(8,0.12)	207.5	≈ 100.0	99.2
pcb3	(4,0.21)	56.6	≈ 100.0	99.4
plants	(11,0.08)	99.5	≈ 100.0	97.0
Ringo 1	(34,0.03)	25.2	≈ 100.0	80.4
Ringo 2	(34,0.03)	28.0	≈ 100.0	81.1
Board 1	(34,0.03)	16.4	≈ 100.0	53.6
Board 2	(34,0.03)	15.6	≈ 100.0	30.4
Board 3	(14,0.06)	78.7	≈ 100.0	95.4
Wafer 1	(12,0.09)	73.5	≈ 100.0	95.3
Wafer 2	(11,0.08)	107.5	≈ 100.0	96.2
Wafer 3	(25,0.04)	23.1	≈ 100.0	53.5

4. Experimental results

In this section we provide experimental results aimed at comparing our algorithm to the FS algorithm based on SAD/SSD and to an FFT-based algorithm. All the compared algorithms were implemented in *C* and run on a Linux workstation based on a *P4 3.056 GHz* processor. Our algorithm includes the run-time method for the estimation of the position of the best matching subwindow and uses as parameters the pair (r, n) yielding the maximum speed-up. The datasets used for the experiments consist of grayscale images and templates of various dimensions and are shown in Figg. 2 - 7.

Table 1 shows the speed-ups (i.e. ratios of measured execution times) obtained by our algorithm with respect to the FS SSD-based algorithm, while Table 2 shows the speed-ups in the case of the SAD function. Both tables show in the second column the values of parameters (r, n) yielding the maximum speed-up, which is reported in column 3. Finally, columns 4 and 5 show the percentage of positions that were pruned thanks to the application of, respectively, all the sufficient conditions and only the first sufficient condition (i.e. inequality (9)).

As it can be noted from the two tables, the proposed algorithm yields significant speed-ups with respect to the FS algorithm throughout the whole dataset. In the SSD case the speed-up ranges from 15.6 (*Board 2*) up to 207.5 (*paint3*),

Figure 2. Datasets *paint 1, 2* and *3*.

Figure 3. Dataset *plants*.

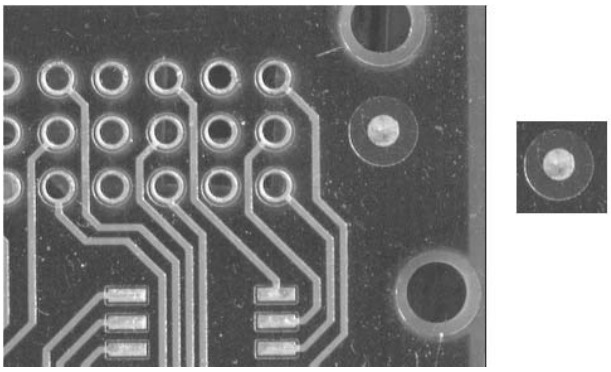

Figure 4. Dataset *pcb3*.

Proceedings of the IEEE International Conference
on Video and Signal Based Surveillance (AVSS'06)
0-7695-2688-8/06 $20.00 © 2006 IEEE

Figure 5. Datasets *ringo 1* and *2*.

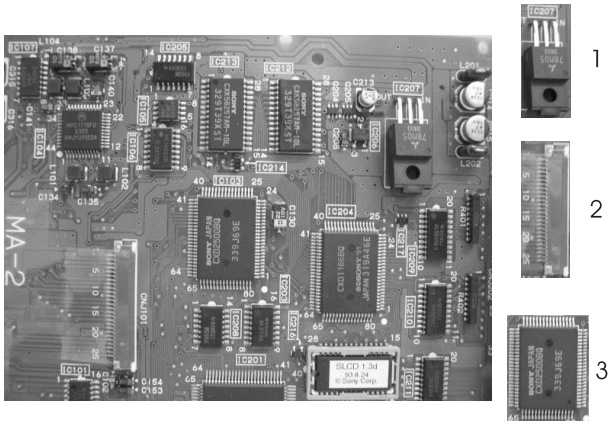

Figure 6. Datasets *Board 1, 2* and *3*.

Figure 7. Datasets *Wafer 1, 2* and *3*.

Table 2. Speed-ups yielded by our algorithm vs. FS algorithm, SAD case.

Dataset	$(r, \frac{n}{N})$	$Speed-up$	$P_{tot}[\%]$	$P_0[\%]$
paint1	(10,0.11)	242.2	≈ 100.00	99.30
paint2	(11,0.09)	206.7	≈ 100.00	99.32
paint3	(7,0.13)	277.0	≈ 100.00	99.82
pcb3	(4,0.22)	73.7	99.99	99.48
plants	(3,0.22)	167.2	≈ 100.00	99.90
Ringo 1	(13,0.08)	44.9	99.98	94.12
Ringo 2	(13,0.08)	68.6	≈ 100.00	95.85
Board 1	(25,0.04)	15.1	$\approx$100.00	74.77
Board 2	(34,0.03)	21.2	≈ 100.00	79.95
Board 3	(7,0.10)	115.9	≈ 100.00	98.71
Wafer 1	(8,0.13)	104.3	≈ 100.00	98.57
Wafer 2	(9,0.09)	139.8	≈ 100.00	99.20
Wafer 3	(16,0.06)	112.5	≈ 100.00	97.20

while it varies between 15.1 (*Board 1*) and 277.0 (*paint3*) in the SAD case.

Finally, we have compared our algorithm to a template matching algorithm based on the SSD function and the FFT. FFT methods are widely used for SSD-based template matching since the SSD function can be written as:

$$\delta_2(x,y) = ||I_s(x,y)||_2 2 + ||T||_2 2 - 2 \cdot I_s(x,y) \circ T \quad (17)$$

with $\circ$ representing the dot product operation. Thus, FFT-based methods compute the dot product term in the frequency domain in order to reduce the overall calculations. In our experiments we have used an efficient algorithm named *cvMatchTemplate*, which belongs to the *OpenCV* computer vision library, developed by *Intel*. Table 3 shows the speed-ups yielded by our algorithm with respect to the *cvMatchTemplate* algorithm for SSD-based template matching. It is worth observing that our algorithm can yield notable speed-ups with regards to the FFT-based algorithm: in the worst case the performance of the two algorithms is the same (speed-up equal to 1.0, *Board 2*), while in the best case the speed-up reaches 12.1 (*Paint3*).

5. Conclusion and future work

A novel method aimed at performing fast and exhaustive template matching based on the L_p norm has been described throughout the paper. The method relies on a partitioning scheme applied to both the template and the image subwindow, as well as on the combined use of the triangular inequality together with partial evaluations of the L_p norm. This approach permits to determine several lower bounding functions, each one representing a closer approx-

Proceedings of the IEEE International Conference
on Video and Signal Based Surveillance (AVSS'06)
0-7695-2688-8/06 $20.00 © 2006 IEEE

Table 3. Speed-up yielded by our algorithm vs. FFT-based algorithm, SSD case

Dataset	(W, H)	(M, N)	$Speed-up$
paint1	$(1152, 864)$	$(164, 161)$	2.8
paint2	$(1152, 864)$	$(128, 152)$	7.2
paint3	$(1152, 864)$	$(118, 162)$	2.1
pcb3	$(384, 288)$	$(72, 73)$	7.4
plants	$(512, 400)$	$(104, 121)$	9.0
Ringo 1	$(640, 480)$	$(126, 144)$	2.1
Ringo 2	$(640, 480)$	$(118, 162)$	2.2
Board 1	$(640, 480)$	$(63, 179)$	1.3
Board 2	$(640, 480)$	$(106, 138)$	1.0
Board 3	$(640, 480)$	$(65, 149)$	5.5
Wafer 1	$(640, 480)$	$(119, 84)$	7.1
Wafer 2	$(640, 480)$	$(109, 123)$	6.8
Wafer 3	$(640, 480)$	$(189, 98)$	1.2

imation of the distance measure used in the template matching process. It has been shown that the computation of the bounding functions is particularly efficient and requires low memory usage. The experimental results show notable speed-ups when comparing the proposed algorithm to the FS algorithm, as well as to an efficient implementation of the FFT-based template matching approach.

Our future work will include the extension of the concept at the basis of the proposed method to the *Block Matching* field for motion estimation. Moreover, we plan to compare our method with the recent approach based on the Walsh-Hadamard transform proposed in [4].

A Proof of inequality (16)

For the sake of brevity we consider here only the case $p = 2$. Suppose to partition vectors $I_s(x, y), T$ into two sub-vectors respectively of rows $[1, \rho]$ and $[\rho+1, N]$ (hence, $r = 2$). We define:

$$I_1 = \sum_{i=1}^{M}\sum_{j=1}^{\rho} I(x+i, y+j)^2 \quad (18)$$

$$I_2 = \sum_{i=1}^{M}\sum_{j=\rho+1}^{N} I(x+i, y+j)^2 \quad (19)$$

$$T_1 = \sum_{i=1}^{M}\sum_{j=1}^{\rho} T(i, j)^2 \quad (20)$$

$$T_2 = \sum_{i=1}^{M}\sum_{j=\rho+1}^{N} T(i, j)^2 \quad (21)$$

hence:

$$\beta_2(x, y) = (\sqrt{I_1 + I_2} - \sqrt{T_1 + T_2})^2 \quad (22)$$

$$\beta_{2,2}(x, y, \rho)|_1^N = (\sqrt{I_1} - \sqrt{T_1})^2 + (\sqrt{I_2} - \sqrt{T_2})^2 \quad (23)$$

Let's assume that:

$$\beta_2(x, y) > \beta_{2,2}(x, y, \rho)|_1^N \quad (24)$$

By using (22) and (23) and by algebraically manipulating (24) we get to:

$$\sqrt{(I_1 + I_2) \cdot (T_1 + T_2)} < \sqrt{I_1 \cdot T_1} + \sqrt{I_2 \cdot T_2} \quad (25)$$

Then, by squaring (25) and by means of simple manipulations we arrive at:

$$I_2 \cdot T_1 + I_1 \cdot T_2 - 2 \cdot \sqrt{I_2 \cdot T_1 \cdot I_1 \cdot T_2} = (\sqrt{I_1 \cdot T_2} - \sqrt{I_2 \cdot T_1})^2 < 0 \quad (26)$$

which is an absurd result. Hence:

$$\beta_2(x, y) \leq \beta_{2,2}(x, y, \rho)|_1^N \quad (27)$$

The proof in general case of r partitions can be obtained straightforwardly by successive applications of (27).

References

[1] D. Barnea and H. Silverman. A class of algorithms for digital image registration. *IEEE Tran. on Computers*, C-21(2):179–186, 1972.

[2] X. Gao, C. Duanmu, and C. Zou. A multilevel successive elimination algorithm for block matching motion estimation. *IEEE Tran. Image Processing*, 9(3):501–504, 2000.

[3] M. Gharavi-Alkhansari. A fast globally optimal algorithm for template matching using low-resolution pruning. *IEEE Tran. Image Processing*, 10(4):526–533, 2001.

[4] Y. Hel-Or and H. Hel-Or. Real-time pattern matching using projection kernels. *IEEE Tran. Pattern Analysis and Machine Intelligence*, 27(9):1430–1445, 2005.

[5] C. Lee and L. Chen. A fast motion estimation algorithm based on the block sum pyramid. *IEEE Tran. Image Processing*, 6(11):1587–1591, 1997.

[6] W. Li and E. Salari. Successive elimination algorithm for motion estimation. *IEEE Trans. on Image Processing*, 4(1):105–107, 1995.

[7] M. Mc Donnel. Box-filtering techniques. *Computer Graphics and Image Processing*, 17:65–70, 1981.

[8] G. Vanderburg and A. Rosenfeld. Two-stage template matching. *IEEE Trans. on Image Processing*, 26:384–393, 1977.

[9] H. Wang and R. Mersereau. Fast algorithms for the estimation of motion vectors. *IEEE Trans. on Image Processing*, 8(3):435–439, 1999.

[10] B. Zitová and J. Flusser. Image registration methods: a survey. *Image and Vision Computing*, 21(11):977–1000, 2003.

Proceedings of the IEEE International Conference
on Video and Signal Based Surveillance (AVSS'06)
0-7695-2688-8/06 $20.00 © 2006 IEEE