Correspondence using Mahalanobis Distance, Dynamic Programming, and Relaxation.

Nicolas Merlet
Institute of Computer Science
The Hebrew University of Jerusalem
91904 Jerusalem, Israel
E-mail : nicolas@cs.huji.ac.il

Abstract

We present a new solution to the correspondence problem. It is based on dynamic programming, and can deal with image sequences.
The overall potential characterizes the quality of matching of a point, and dynamic programming achieves consistency independently for each edge of the new image. The overall potential is defined on a *pair of points* as a function of two components : a potential defined for each new *point* as a function of its Mahalanobis distance to the old edges, and a potential associated to their respective *edges* on the basis of their relative distance and orientation.
Finally, occlusions, artifacts and ambiguities are dealt with by applying relaxation to the different solutions found by dynamic programming, where the compatibility coefficients associated to the empty label are updated at each iteration.

1 Introduction

The *correspondence problem* is fundamental in stereo vision : how to match pixels or other primitives between two given images in an optimal way. We generalize this problem to the matching between a new image and a probability distribution obtained from an image sequence.

Dynamic programming is a successful method of optimization. The basic approach consists in defining a cost or potential depending on the local characteristics of the image and to perform an iterative summation-minimization process. The absence of any threshold is an important factor of robustness.

In particular, Baker [1] and Ohta [5] showed how to apply it to the correspondence problem. They first rectify the pair of stereo images so that the epipolar lines are parallel to the scan-lines. A 2D array is then built for a couple of corresponding lines, and the cost measures the similarity between corresponding features, grey-levels, and intervals. The consistency between successive lines is achieved either with a cooperative process to correct errors [1] or by searching a 3D space [5].

In our solution, the overall potential characterizes the quality of matching of a point and its sum is computed along each edge. This potential has two components. We first assume that we know for each *point* of the first image the corresponding probability distribution in the plane of the new image (a distance function may be sufficient), and present an algorithm to compute the map of Mahalanobis distances w.r.t. the edges. A second potential is defined between *edges* in the first image on the basis of their relative distance and orientation. We define the overall potential on a *pair of points* as a function of the points probabilities and of the potential between their respective edges.

Global matching is performed before dynamic programming by finding the offset between the probability distribution and the new edges. While dynamic programming achieves consistency independently for each edge of the new image, relaxation achieves consistency between the edges : ambiguities or occlusions remaining after dynamic programming are solved by applying a relaxation process to the different solutions obtained. The coefficients associated to the empty label are updated through the iterations.

We discuss finally the complexity and the quality of the results, which show the adequacy of the method to active vision systems and to image sequences.

2 Overview

Let us recall that for a random vector x with normal distribution of average u and variance Σ, the Mahalanobis distance is defined as : $d_m(x,u) = (x-u)^t.\Sigma^{-1}.(x-u)$. The sets of constant d_m, $\{x/d_m(x,u) = c\}$, define hyper-ellipsoids centered on u, whose axes and lengths are given by the eigenvectors and eigenvalues of Σ.

We assume that we know the displacement of the camera between two images I_1 and I_2. The probability distributions in 2D or 3D are supposed normal, but the method may be applied with other distributions as well. In order to find the matching between the edges, we perform the following steps :

1. we *find the edges* E_1 and E_2 of both images by performing a Deriche filtering. We group together the points close to the same line and label the edges for both images. We order the points w.r.t. the distance to the extremities of each edge and eventually we subsample the points of E_1 to gain time

2. we then build the *probability map* : if only one image has been taken, we project the 2D points of E_1 and their covariance matrices in 3D (we assume a uniform depth), and project them back on the plane of I_2. If an image sequence is used, only the projection from 3D to 2D is performed. We compute maps of d_m w.r.t. each edge and a global map w.r.t. the union of the edges (Section 3).

3. we use this global map for performing *global matching* : we first compute the 2D translation so that the mass centers of E_2 and of the projected edges of E_1 fit. Then, we compute the sum of the probabilities for all the points of E_2 and update the translation by one until a local maximum is found.

4. we then perform *edge matching* : we first compute the potential matrix between the edges in E_1 or in 3D (Section 4.1). With this matrix and the probability potential (Section 4.2), we compute the overall potential, and we find which portions of which edges of E_1 fit to each edge of E_2 by looking for the shortest path with Dynamic Programming (Section 4.3).

5. in order to solve ambiguities, we store all the best solutions obtained by dynamic programming for each new edge, and we apply *relaxation* by considering each solution as a possible label associated to an edge (Section 5).

3 Mahalanobis map

We generalize the definition of d_m to sets of points similarly to the usual distance. We first compute for each *edge* of E_1, the 2D map of d_m from each pixel of I_2 to the projected edge. The probability potential is a function of this series of maps. A second structure is one 2D map obtained as the minimum of these maps. It characterizes d_m from each pixel of I_2 to the *union* of projected edges, and is used for global matching.

The problem is that the probability is too low in a great part of the surface for being of any interest and relevant information is to be found near the edges only. We therefore define for each edge an interest window. Since up to 50% of the surface of the window is still irrelevant, we define a threshold under which the probability is neglected. We then propagate from the edge points until the limit of the (approximated) ellipse associated to the edge and to the threshold.

Instead of propagating the distance values [2], we propagate the label of the closest edge point and compute directly d_m with respect to it. Initially we mark in the 2D surface the pixels corresponding to the measurements of the edge points with their own label. The other pixels are null pointers. Then we perform two scannings (top-left to bottom-right and backwards), with the updating equation (Fig. 1, left) : If (x_i, y_i) is marked with a label l_i of an edge point,

- for each of its 4 neighbors (x_j, y_j), we compute d_m from (x_j, y_j) to the point l_i w.r.t. the covariance matrix of l_i
- if the distance is below a given threshold t_d, then if (x_j, y_j) is not labeled or if the new value of the distance is smaller than the previous one, we update (l_j, d_j) : l_j is l_i and d_j is the new distance computed.

This transformation does not preserve rotation as Fig. 1 (middle) shows : N_8 gets a label, while M_2 does not ; no path may be build between M_0 and M_2 inside the ellipse and in two scannings. We solve this problem with an approximate solution by alternating systematically the direction of scanning at each new line.

4 Dynamic programming

4.1 Computing the edge potential

We first compute the distance $d_{i,j}$ between edges e_i and e_j as the minimum euclidean distance between the sets $\{P_{i,1}, P_{i,2}\}$ and $\{P_{j,1}, P_{j,2}\}$, where $P_{i,1}$ and $P_{i,2}$ are the

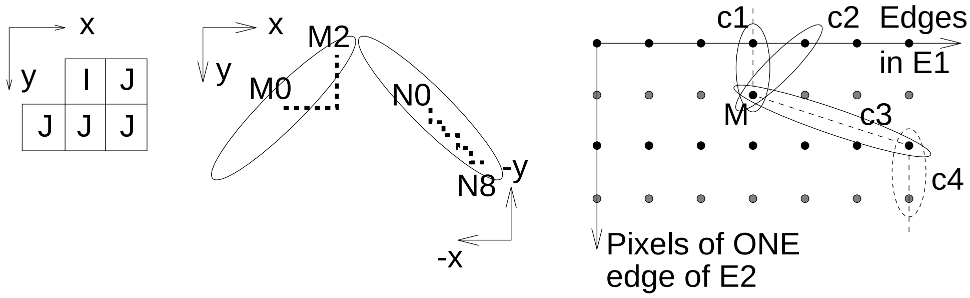


Figure 1: Left : Mask for the distance. Middle : anisotropic propagation. Right : pairs of points and neighborhood for dynamic programming

extremities of e_i. Assuming that $d_{i,j} = d(P_{i,1}, P_{j,1})$, we then define the angle $\theta i, j$ as the angle between the vectors $P_{i,2}\vec{P}_{i,1}$ and $P_{j,1}\vec{P}_{j,2}$. We define the potential $c(e_i, e_j)$ as a linear combination of $d_{i,j}$ and $\theta i, j$, such that it is null for edges prolongating one an other. The potential increases with the deviation to this configuration and is maximum for parallel edges.

4.2 Computing the probability potential

We define a 3D volume in the following way. z is the axis of edges in E_2 and each plane of constant z (Fig. 1, right) corresponds to one edge. y is the axis of the pixels of this edge and each row corresponds to one pixel. x is the axis of edges in E_1 and each column corresponds to one edge.

As in previous works [4], we define the potential on *sets* of neighboring points. Now, the neighborhood of a point of the row y is defined as the set of the points of the rows y-1 and y+1, and we define the potential on a pair of points. In Fig. 1 (right) for instance, the neighbors of M are the points in black, and c_1, c_2 and c_3 are possible pairs containing M. A possible configuration of matching for the edge e_i of E_2 is represented as a dashed line ; it goes through two different old edges (columns), and e_i is matched with portions of these edges.

We scan independently each level. For each point p_i of an edge of E_2, we find the edges e_j of the first image E_1 for which the thresholded probability is not null, by comparing with the coordinates of the windows (Section 3). For each such edge e_j, we store the opposite of the value of the probability at the location p_i (the potential pot_p should be low for high probabilities) as well as the label of the corresponding point of e_j (closest w.r.t. d_m).

4.3 Search of shortest path

We then perform dynamic programming in a similar 3D structure :

- we initiate the row of ordinate zero (corresponding to the extremity of the edge e_i of E_2), for edges e_j of E_1 where the probability is not null, with the label of the pixel of e_j (in order to recover the matching of pixels), the probability potential (in order to propagate it to the next row), and the label of the edge e_j, useful during back-propagation.

- we then perform one scanning ; for each row k, starting from 0, for each edge $e_{j,0}$ where the probability is not null ; we look for the edges e_j for which the probability at row (k+1) is not null ; for each such edge, we update the energy $U(j,k+1,i)$ with the sum $U(j_0,k,i)+c(e_j,e_{j_0}).L_i+\frac{pot_p(j_0,k,i)+pot_p(j,k+1,i)}{2}$, and store the label of the edge point of e_j and the label of the edge e_{j_0}, if U decreases. We use the length L_i of edge i to normalize the weights of the partial potentials.

- we find on the last row the minimum value of U for edge $e_{j,0}$ e.g. We propagate backward from this point, following the edge labels which we have stored. Each point of a new edge e_i is associated with a point of an old edge e_j.

5 Relaxation

Several problems remain : first, artifacts such as highlights or shadows should be eliminated. Second, if the edges are close to another or if a new edge appears, then two new edges may be associated to the same old edge (occlusion problem).

Let us suppose that we have obtained for each edge e_i a set of possible matchings (labels) $\{\lambda_1,...,\lambda_m\}$ where $\lambda_l=\{M_1,...,M_p\}$ is a set of 2D/3D pts. We want to keep for each edge only the label λ_l having the minimum of conflicts with other edges, through relaxation labeling [3]. Each edge may be matched with the empty path. For each label λ_l of each edge e_i, we define an initial probability $p^0(\theta_i=\lambda_l)$. A compatibility coefficient $r(\theta_i=\lambda_l,\theta_j=\lambda_k)$ is defined between each couple of matchings (e_i,λ_l) and (e_j,λ_k). The support for a given label λ_l is then defined at each iteration n by $Q^n(\theta_i=\lambda_l)=\Sigma_{j=1}^N C_{ij}\Sigma_{k=1}^m r(\theta_i=\lambda_l,\theta_j=\lambda_k).p^n(\theta_j=\lambda_k)$ and the probability is updated according to : $p^{n+1}(\theta_i=\lambda_l)=\frac{p^n(\theta_i=\lambda_l).Q^n(\theta_i=\lambda_l)}{\Sigma_{k=1}^m p^n(\theta_i=\lambda_k).Q^n(\theta_i=\lambda_k)}$. The aim of relaxation is generally to obtain more significant probabilities (closer to one or to zero) but the theoretic issues of convergence and of choice of the compatibility coefficients are not entirely solved.

5.1 Computing all the best paths

We showed in Section 4 how to find for each edge e_i the shortest path of matching points. We find all the best paths in a similar way. Let us call $S_i=\{e_{j1},...,e_{jq}\}$ the set of all the edges of E_1 for which the probability in the plane defined by e_i has non-null values (Fig. 4), and $P(S_i)$ the set of combinations of S_i. We scan the sets s_j of $P(S_i)$ in decreasing order of number of elements. For each s_j we find the shortest path by dynamic programming. The search is simplified in two ways : first, if no path is found for s_j, there is no path for any subset of s_j. Second, if the shortest path found for s_j goes only through edges forming a strict subset s_l of s_j, then the same solution is found for any s_k such that $s_l\subset s_k\subset s_j$.

5.2 Choice of the parameters

$p^0(\theta_i=\lambda_l)$ is defined for each edge e_i and each shortest path λ_l, inversely proportional to the score of the path : the higher the score, the lower the probability. The empty path is given a constant non-null initial probability $p_0<1$.

We define as neighborhood of a new edge e_i all the new edges e_j associated to a common old edge for one of their shortest paths, and the weights C_{ij} as the length L_j of edge e_j.

For non empty paths, we define the compatibility $r(\theta_i = \lambda_l, \theta_j = \lambda_k)$ for two edges e_i and e_j and their paths λ_l and λ_k inversely proportional to the variance of the disparity of the set of their points taken together. The coefficients $r(\theta_i = \lambda_l, \theta_j = \emptyset)$ are taken null since no information may be deduced about e_i when e_j is not matched, and so is $r(\theta_i = \emptyset, \theta_j = \emptyset)$.

We define the compatibility coefficient $r(\theta_i = \emptyset, \theta_j = \lambda_{j0}) = r_1.r_2$. r_1 is the minimum value of the coefficient for enforcing the increase of $p^n(\theta_i = \emptyset)$ and is equal to $\frac{\Sigma_{l \neq \emptyset} p^n(\theta_i=\lambda_l) \Sigma_j L_j \Sigma_{k \neq \emptyset} r(\theta_i=\lambda_l, \theta_j=\lambda_k).p^n(\theta_j=\lambda_k)}{L_{j0}.(1-p^n(\theta_i=\emptyset))}$. r_2 is the criterion of incompatibility between e_i and the matching of e_j with λ_k and is equal to $\Sigma_{l \neq \emptyset} p^n(\theta_i = \lambda_l).com(i, \lambda_l, j, \lambda_k).c(e_i, e_j)$ where $com(i, \lambda_l, j, \lambda_k)$ is the percentage of common points for this configuration of matching and $c(e_i, e_j)$ is a cost between edges similar as the one used in Section 4.1. These coefficients only are updated at each iteration according to the new probabilities, in order to enforce the discrimination of artifacts.

Convergence is practically obtained after a constant number of iterations (50).

6 Discussion

Fig. 2 shows the original images of a first example (left, middle). The right image represents the edges of the new image, superimposed on the projected probability distribution. The two windows correspond to two edges of E_2 taken as examples.

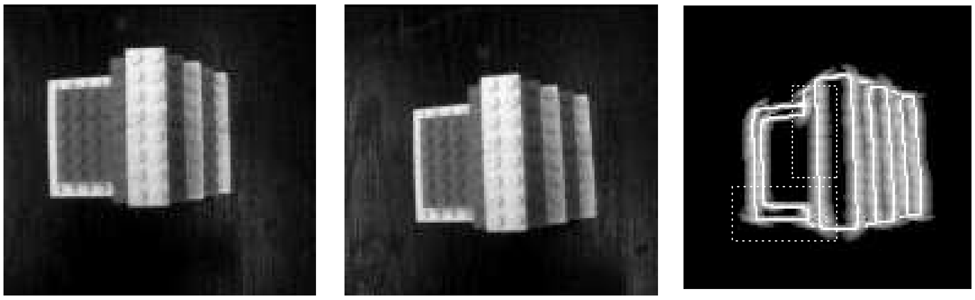


Figure 2: Original images (left, middle). Superposition of E_2 and of d_m (right).

In Fig. 3, we have labeled these two edges of E_2 (right) as well as the edges of E_1 (left) which have been found close enough w.r.t. d_m to them.

Figure 3: Labeled edges. Left : edges of E_1. Right : two edges of E_2

Fig. 4 shows the two levels (top and bottom) of the 3D structure of the potential (left) corresponding to the two example edges. The old image contains 27 edges. In the first window, the closest edges (20,21,23) correspond to the columns with the lowest potential (top-left). In the second window (bottom-left), the closest edges have the labels 1 and 17. The energy obtained by integrating the potential (downwards) is presented in Fig. 4 (middle). The energy is low for the edges mentioned previously. Fig. 4 (right) shows the paths obtained during back-propagation. These paths are continuous *with respect to the neighborhood* which we have defined.

Figure 4: Potential, energy, and paths for edge 25 (top) and 10 (bottom)

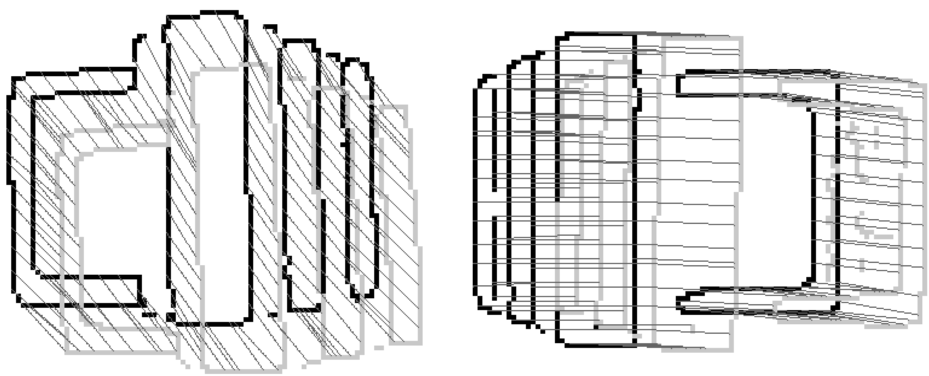
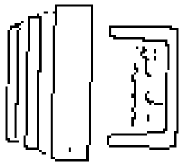


Figure 5: Correspondence for the images of Fig. 2 (left) and of Fig. 6 (right)

Fig. 5 (left) represents the matching obtained for all the edges. Fig. 6 represents the edges of a second example (middle) and the original image I_2 (right). Fig. 5 (right) shows the result, where the shadow and highlights have been eliminated automatically by relaxation. Similar results are obtained when a new edge appears in the new image due to a better resolution. However, highlights coming too close to an edge disturb the edge detection itself.

In the current state of work, the times in seconds for running the different parts on a SUN4 and the corresponding complexities are given in Fig. 6 (left). N_p is the number of 3D points used for the matching, N_e the number of edges, L_e the number of pixels in an edge, and S the surface of a window containing an edge. All

Operation	Time	Complex
Proj. 3D/2D	0.2	Np
Proba. map	7.0	SxLe
Edg. Pot.	0.3	NexNe
Prob. pot.	0.12	NexNexLe
Dyn. Prog.	0.27	NexNexLe
Relaxation	1.2	Ne



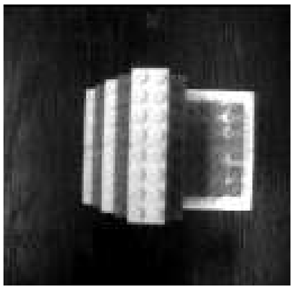


Figure 6: Complexity (left). Edges (middle) and image I_2 (right) of example 2.

the experiments were performed with 128x120 images. The essential of the work (dynamic programming and relaxation) is done in fractions of seconds. Current research involves computing only the part of the Mahalanobis map really needed, or using a smaller resolution, or approximating the distance [2].

7 Conclusion

We have presented a new approach for performing matching of edge points, with dynamic programming. We focus the constraints on the consistency inside each edge : each point should be associated to a point with small Mahalanobis distance, the edges associated should verify a proximity criterion (distance and orientation) between them, and the number of changes of matchings to these edges is limited. That is, dynamic programming favors "long runs", successive pixels of one edge should be associated to successive pixels of the same edge.

We solve ambiguities with relaxation, and deal with the empty label through new time-varying compatibility coefficients. The 3D structure used during the computations gives also a clear representation and detection of the occlusions. The formalism presented allows to solve the correspondence problem for image sequences.

References

[1] H.H. Baker, T.O. Binford, Depth from edge- and intensity- based stereo. Proceedings 7th Joint Conf. on Art. Int., pp. 631-636, 1981.

[2] G. Borgefors, Distance Transformations in Arbitrary Dimensions, *CVGIP 27*, pp. 321-345, 1984.

[3] J. Kittler, J. Illingworth, Relaxation labelling algorithms - a review, *Image and Vision Comp.*, vol. 3-4, pp. 206-216, 1985.

[4] N. Merlet, J. Zerubia, A curvature-dependent energy function for detecting lines in satellite images, *8th SCIA Conference*, Tromso, 25th-28th May, 1993.

[5] Y. Ohta, T. Kanade, Stereo by Intra- and Inter-Scanline Search Using Dynamic Programming, *T-PAMI*, vol. 7-2, pp. 139-154, 1985.