On the Problem of Correspondence in Range Data and Some Inelastic Uses for Elastic Nets

Anupam Joshi, *Member, IEEE,* and Chia-Hoang Lee

***Abstract*—In this work, the authors propose a novel method to obtain correspondence between range data across image frames using neural like mechanisms. The method is computationally efficient and tolerant of noise and missing points. Elastic nets, which evolved out of research into mechanisms to establish ordered neural projections between structures of similar geometry, are used to cast correspondence as an optimization problem. This formulation is then used to obtain approximations to the motion parameters under the assumption of rigidity (inelasticity). These parameters can be used to recover correspondence. Experimental results are presented to establish the veracity of the scheme and the method is compared to earlier attempts in this direction.**

I. Introduction

Correspondence is defined by Ullman [32] as the process by which elements in different views are identified as representing the same object at different times, thereby maintaining the perceptual identity of objects in motion. It can be said sans hesitation that the problem of obtaining correspondence is a fundamental aspect of computational vision and underlies much work on motion. The various approaches to the measurement of visual motion can be broadly categorized as relying either on optical flow techniques or on feature based techniques. It is with the latter that we concern ourselves in this work.

Feature based methods establish correspondence between feature points (or tokens obtained from the raw image data) and use these correspondences to obtain the parameters that describe the motion in the image sequence. Establishing the correspondence is clearly a prerequisite to further processing in feature based schemes. Many efforts in the area of dynamic image analysis, however, assume that this underlying problem of correspondence has been resolved [1], [2], [5], [13], [16], [17], [23], [25], [28], [31], [33], and [35].

While the objects in the real world are three dimensional, research in the area of correspondence has dealt mostly with two dimensional images [4], [18]–[20], [26], [27], [29], [34], and [37]. With the increasing availability of equipment to do range sensing, however, the problem of establishing correspondence between range data, the three dimensional representation of the object, is gaining prominence. Huang and Chen [6] have proposed a scheme that uses preestablished correspondence between three points. Let p_1, p_2, p_3 be points from the first frame, and q_1, q_2, q_3 be their corresponding points in the second frame. Given any point p_i in the first frame and q_j in the second, it can be shown that tetrahedron p_1, p_2, p_3, p_i is congruent to tetrahedron q_1, q_2, q_3, q_j iff points p_i and q_j correspond. In [22] Huang and Lin propose a technique that works very effectively in the absence of noise. They use centroids of the two token sets to obtain two new sets of tokens which are related by rotation only. Let p_1 and p_2 be the two point sets, and let c_1 and c_2 be their centroids, respectively. They obtain token sets q_1 and q_2 by setting

$$q_{1i} = p_{1i} - c_1$$

and

$$q_{2i} = p_{2i} - c_2$$

where the subscript i denotes the ith member of the token set q_1 or q_2. These new point sets are used to get four candidates for the rotation matrix, $\mathbf{R}$. Correspondence is obtained from these by choosing the correct $\mathbf{R}$.

Another technique which can tolerate noise better is proposed in [21]. It involves obtaining a good initial estimate to the rotation axis and uses Fourier transforms, making it computationally expensive. Magee *et al.* [24] have used subgraph matching when the objects in the scene are polyhedral or cylindrical to obtain correspondence in range data. They also propose an interesting method to find suitable "feature points" in the object for which range data is obtained. Some other approaches to this problem can be found in [12], [15], and [30]. Shuster [30] uses a quadratic loss function to obtain an optimal rotation matrix and reduces this problem to finding the optimal quaterion. Faugeras and Herbert [12] use a similar technique applied to the vertices and planes of an object, to match it with a model by obtaining optimal translational and rotational motion parameters that relate the range data with a stored model. This method, however, is not computationally very efficient. To do an image to model match, Grimson and Lozano-Perez [15] use an involved tree pruning approach. Their approach requires knowing the surface normal at each measured point and uses distance and angular constraints to obtain a matching.

In the present work, we propose a simple scheme which uses an elastic net like approach. The proposed method is able to handle missing points and a substantial amount of noise in the data, and is computationally efficient. In the sections that follow, we briefly outline the concept of elastic nets and then expound our method for obtaining correspondence. We also

Manuscript received September 12, 1992; revised September 23, 1993, February 2, 1994 and March 18, 1994. This work was supported in part by the National Science Council of the R.O.C. Grant NSC 82-0408-E-009-366.

A. Joshi is with the Computer Science Department, Purdue University, West Lafayette, IN 47907 USA.

C.-H. Lee is with the Department of Computer and Information Sciences, National Chiao-Tung University, Hsinchu, Taiwan 30050 R.O.C.

IEEE Log Number 9404858.

present the results of extensive simulation with synthesized and real data.

II. Elastic Nets

Durbin and Willshaw, in [10], proposed a novel scheme to solve combinatorial problems that involve geometrical structures and topographical mappings between them. They showed how this method could be used to solve the traveling salesman problem. Their basic concept involves using a deformable contour, which is changed in shape by forces to approximate the optimal valid tour. The forces that change its shape are those that attract the contour points to cities and those that try to keep neighboring points of the contour together. This is akin to stretching a rubber band to make it pass through all the cities to obtain the tour. Durbin and Willshaw show that deforming a contour in this manner is akin to minimizing the energy of the system, which is formulated as

$$\mathcal{E} = -\alpha K \sum_i \ln \sum_j \phi(d_{ij}, K) + \beta \sum_j |y_{j+1} - y_j|^2 \quad (1)$$

where $d_{ij} = |x_i - y_j|$ and $\phi(d, K) = \exp(-d^2/2K^2)$.

The x_i's represent the coordinates of the cities and y_j's represent the coordinate of the points on the contour. They show that if there are more points on the contour than there are cities (in their simulation, the ratio is 2.5), then in the limit that $K \to 0$ a valid, close to optimal tour is produced. Since $\mathcal{E}$ is bounded from below, it requires that as $K \to 0$

$$\forall x_i \, \exists y_j \text{ s.t. } |x_i - y_j| \to 0.$$

This ensures that the contour passes through all cities. Moreover, as the number of points on the rubber band is increased, the second term in the energy function is minimized by placing all points at equal distances from each other. If $\mathcal{D}$ be the total path length, such a configuration makes the value of the second term $\frac{\mathcal{D}^2}{\text{Number of points}}$, which is obviously minimized by reducing the path length.

To obtain the tour then, we merely need to do gradient descent on the energy surface defined by $\mathcal{E}$, which is achieved by updating the positions of the points on the rubber band, y_j, by $K\partial\mathcal{E}/\partial y_j$ at each iteration step. Computing this quantity, we obtain Δy_j, the change in value of y_j at a given iteration as

$$\Delta y_j = \alpha \sum_i w_{ij}(x_i - y_j) + \beta K(y_{j+1} - 2y_j + y_{j-1})$$

where

$$w_{ij} = \frac{\phi(d_{ij}, K)}{\sum_l \phi(d_{il}, K)}.$$

Durbin and Willshaw noted that this approach produced better tours than the Hopfield net, and this method scaled better with the number of cities as well. Readers interested in a detailed theoretical analysis of this are referred to [9] and [36].

III. Method

We now outline how the concept of elastic nets can be used to obtain correspondences. Let A'_i be a point token from the first set and B'_i be one from the second set. We can represent the correspondence by a permutation σ such that the point $B'_{\sigma(i)}$ from the second frame corresponds to the token A'_i in the first frame. Let $\mathbf{R}$ and $\mathbf{T}$ be the rotation and translation, respectively, that define the motion from the first to the second frame. Assuming that the motion is rigid, we get

$$B'_{\sigma(i)} = \mathbf{R}A'_i + \mathbf{T}. \quad (2)$$

As explained in Section I, Huang *et al.* [6] showed that using the centroids, we can transform the point sets A' and B' into A and B such that

$$B_{\sigma(i)} = \mathbf{R}A_i. \quad (3)$$

Let us suppose that some oracle can give us the rotation matrix $\mathbf{R}$. Then, correspondence can be trivially established by observing that if point i corresponds to point j, then $B_j \equiv \mathbf{R}A_i$. If correspondences are unique, then this is a necessary and sufficient condition for establishing them. Suppose that instead of getting $\mathbf{R}$, we get an approximation $\mathbf{R}'$ to it. Correspondence can then be established by observing that $d_{ij} = \min_k d_{kj}$ where d_{ij} is the distance between points B_j and $\mathbf{R}'A_i$.

The use of elastic nets comes in obtaining $\mathbf{R}$. We take the energy function of elastic nets to be the following

$$\mathcal{E} = -\alpha K \sum_i \ln \sum_j \phi(d_{ij}, K) \quad (4)$$

where $\phi(d_{ij}, K) = e^{\frac{-d_{ij}^2}{2K^2}}$.

The distance d_{ij} is used slightly differently here. In the original work of Durbin and Willshaw, each point on the contour could move independently of the others, and so distances between the current position of the contour and points representing the cities are computed directly. Since we assume the motion to be rigid, we do not move points individually from the first frame to the second. We merely look for the parameter of motion that would do it. Thus, we define d_{ij} to be $|B_j - RA_i|$. Observe that (4) is simply the first term of the energy function (1) that was proposed in [10], with a different definition of the distance. The minimization of this term ensures that

$$\lim K \to 0, \forall A_i \,, \exists B_j \text{ s.t. } d_{ij} \to 0. \quad (5)$$

In other words, for every point in the first frame, there is a corresponding point in the second frame. The second term of the energy function (1) is used to minimize the path length of the tour and is of no consequence in this problem. This approach is similar to that of deformable templates proposed by Yuille [36].

Now that we have formulated the energy of the system, we can essentially do a gradient descent to obtain its minimum. The parameter that changes here is the 3×3 rotation matrix, R. The rotation of a body in three space is defined in terms of its axis of rotation and the angle by which it is rotated about this axis. Its structure is given by the equation shown at the bottom of the page, where n_1, n_2, and n_3 are the direction cosines of the axis of rotation which are determined by its tilt and slant angles, and θ is the angle of rotation. The direction cosines are in turn constrained by

$$n_1^2 + n_2^2 + n_3^2 = 1.$$

Thus there are only three free parameters, namely two of the direction cosines and the angle of rotation. It is also evident that the nine elements of the rotation matrix are related nonlinearly to each other. Some previous researchers have tried to get around this by describing the motion in terms of the quaterion [12], [30]. We chose to treat all nine entries as independent, however, since our aim is merely to obtain correspondence, not an extremely accurate computation of the motion parameters. This reduces the amount of computation involved in obtaining the gradient, $\partial\mathcal{E}/\partial r_{ij}$.

Ideally, if we carried out sufficient iterations, the values of the r_{ij}'s would converge. Since our aim is only to establish correspondence, however, we can save on computation time. This is done by performing a few iterations and then using the nearest neighbor criterion to select the corresponding points.

In principle, we could have defined the distance metric as $|B_j - (RA_i + T)|$ and then computed both the rotation and translation parameters. Using Huang's approach, however, leads to fewer computations while doing the gradient descent.

We now briefly comment upon the computational complexity of the algorithm. In Fig. 1, we show the pseudocode corresponding to the algorithm. The main program essentially involves various initialization procedures and then a fixed number of update cycles, where the entries of R_{ij} are updated using the method described earlier. The update procedure uses a double loop to compute the updated values. The process of updating can be done in constant time. The whole update sequence is thus $O(n^2)$, where n is the number of points. Applying the transformation to the points takes $O(n)$ time and selecting the nearest neighbors can be accomplished in $O(n^2)$ time. The whole algorithm thus takes $O(n^2)$ time.

Having outlined the basic method, we now look at an interesting variation. When multiple frames of the object in motion are obtained, the phenomenon of occlusion often occurs. In occlusion, as the name suggests, the object in motion is partially hidden from view. As such, some frames now contain fewer points than others. For any two frames between which correspondence is to be established, we assume without loss of generality that the first frame contains fewer points. While occlusion is perhaps the norm in real image sequences, most algorithms for correspondence in three-dimensional (3-D) do not address it at all, with [6] being an exception. Our algorithm, however, is ideally suited to this task. Recall that in defining the energy of the system, we did not assume that the number of points in the two frames were the same. In fact, the energy function was constructed to ensure that each point in the first frame found a match in the second. If the two frames have the same number of points, this assures us of a unique match. If the second frame has more points, then we find correct matches for the points in the first frame, and the "extra" points in the second frame are of no consequence.

```
main()
{
    initializeMatrix();
    repeat
        update();
    until (hundred iterations);
    applyComputedRotation();
    selectNeighbours();
}
update()
{
    for j in points of second frame {
        for i in points of first frame
            compute the updates of R_ij;
        update the entries of R_ij;
    }
}
```

Fig. 1. Pseudocode for the algorithm.

We should point out here that we are considering a somewhat restricted version of the missing points problem. There may be points missing from both the frames such that the total number of points in the two frames is the same.

IV. Results of the Simulation

To verify the proposed technique and to determine whether our simplifications hold well under experimental conditions, we performed several simulations. In line with [6], data were generated as points in a cube of side two hundred units with the origin as one of the corner points. The x, y, and z coordinates of the points were chosen as independent random numbers. The data for the second frame was obtained by applying a rotation and translation to the points of the first frame.

The algorithm outlined in the previous section was applied to a large number of data sets, which covered a wide range of motion parameters from small to large. We may point out here that in our implementation, we chose to obtain the rotation matrix $\mathbf{Q}$ that moved the second frame to the first. Note that if $\mathbf{R}$ be the rotation applied to the first frame to

$$\begin{bmatrix} n_1^2 + (1 - n_1^2)\cos\theta & n_1 n_2(1 - \cos\theta) - n_3 \sin\theta & n_1 n_3(1 - \cos\theta) + n_2 \sin\theta \\ n_1 n_2(1 - \cos\theta) + n_3 \sin\theta & n_2^2 + (1 - n_2^2)\cos\theta & n_2 n_3(1 - \cos\theta) - n_1 \sin\theta \\ n_1 n_3(1 - \cos\theta) - n_2 \sin\theta & n_2 n_3(1 - \cos\theta) + n_1 \sin\theta & n_3^2 + (1 - n_3^2)\cos\theta \end{bmatrix}$$

obtain the second, then $\mathbf{Q} = \mathbf{R}^T$. Computing $\mathbf{Q}$ is thus as good as computing $\mathbf{R}$. As an initial approximation, the rotation matrix was set to the identity matrix. It was observed that this gross initial approximation sufficed, in all but a few cases, to obtain correspondence. Also, fewer than 100 iterations were needed to obtain an approximation to $\mathbf{R}$ sufficient to establish correspondence. In Table I, we present the actual rotation matrix as well as the approximation obtained by our method after a hundred iterations for two different instances, a and b. The motion parameters used to obtain the second frame from the first are as follows. For case "a," the translation vector was $[100, 100, 100]^T$ and the tilt, slant, and rotation angles were 20 degrees, 25 degrees, and 50 degrees, respectively. For case "b," the translation vector was $[200, 100, 50]^T$ and the angles were 35 degrees, 45 degrees, and 20 degrees. These examples had ten points in each frame, and all were correctly matched by our method, with the average distance between matched points being 1.369 and 3.071 in the two cases, respectively. For case "a," the angles of tilt, slant, and rotation obtained from the matrix were 15.34 degrees, 24.75 degrees, and 50.124 degrees, and the computed direction cosines were (0.403794, 0.110785, 0.908117). The actual direction cosines were (0.397130, 0.144543, 0.906309), and the angle between the actual and computed axis of rotation was 1.39 degrees. For case "b," the angles obtained were 40.56 degrees, 45.41 degrees, and 23.392 degrees, and the computed direction cosines were (0.541018, 0.463144, 0.701995). The actual values for the direction cosines were (0.579227, 0.405576, 0.707110), and the angle between the actual and computed rotation axis was 3.98 degrees. The algorithm was applied to many such data sets, and the results are summarized in Table II. The first table shows the means and standard deviations of the error in the computed angle of rotation, the number of points matched, and the average distance between the matched points. The figures are from 50 runs of the algorithm, each run having tilt, slant, and rotation values chosen randomly in [0 degrees, 90 degrees] and having 25 points in each frame. The second table shows similar data for 100 runs of the algorithm with 10 points per frame and tilt, slant, and rotation chosen randomly in [0 degrees, 50 degrees]. As remarked earlier, in a few cases, the algorithm did not compute the rotation matrix closely enough within a hundred iterations. This occurs when $\mathbf{I}$ does not serve as a good enough initial approximation for $\mathbf{R}$. In such cases the method converged if the initial estimate was better chosen. In Table III we show the actual $\mathbf{R}$ which could not be approximated starting from $\mathbf{I}$ as well as the initial estimate which caused the method to succeed. Note that good initial estimates can be obtained by Huang's method [22] among others.

TABLE I

$$\mathbf{R}_{act} = \begin{bmatrix} 0.699127 & -0.673766 & 0.239293 \\ 0.714755 & 0.650254 & -0.257423 \\ 0.017842 & 0.351012 & 0.936201 \end{bmatrix}$$

$$\mathbf{R}_{cmp} = \begin{bmatrix} 0.699641 & -0.642314 & 0.232284 \\ 0.708223 & 0.645532 & -0.254673 \\ 0.018761 & 0.340672 & 0.937081 \end{bmatrix}$$

case(a)

$$\mathbf{R}_{act} = \begin{bmatrix} 0.959926 & -0.227677 & 0.163415 \\ 0.256012 & 0.949613 & -0.180811 \\ -0.114014 & 0.215401 & 0.969847 \end{bmatrix}$$

$$\mathbf{R}_{cmp} = \begin{bmatrix} 0.941864 & -0.169072 & 0.164636 \\ 0.256890 & 0.935436 & -0.177320 \\ -0.079931 & 0.108917 & 0.958311 \end{bmatrix}$$

case(b)

TABLE II
SUMMARY OF RESULTS FROM SYNTHESIZED DATA

	Error in θ	No. of Points matched	Average Distance
Mean	6.5299°	21.7241	7.5850
Std. Deviation	4.9846°	8.2415	11.2617

	Error in θ	No. of Points matched	Average Distance
Mean	3.7110°	10	1.3522
Std. Deviation	3.0226°	0	0.0552

TABLE III

$$\mathbf{R}_{act} = \begin{bmatrix} 0.543991 & -0.715635 & 0.438110 \\ 0.823867 & 0.356526 & -0.440604 \\ 0.159114 & 0.600629 & 0.783535 \end{bmatrix}$$

$$\mathbf{R}_{init-est} = \begin{bmatrix} 0.5 & -0.5 & 0.5 \\ 1.0 & 0.2 & -0.5 \\ 0.0 & 0.5 & 1.0 \end{bmatrix}$$

In real world situations, the data are often distorted by noise. So the algorithm was next tested with noisy data to evaluate its ability to operate in a real environment. The errors in data were modeled as zero mean additive noise. Many simulations, with different motion parameters, as well as added noise of different variances, were performed. The results demonstrate that the method was able to handle noise with variance of about 50 units with negligible degradation of performance. This variance is about 25% of the edge length of the cube in which data points were chosen. Performance was degraded, however, for larger variances and the method became unreliable for variations beyond 100 units. Fig. 2 is a graph of the average (over 50 runs) of the number of points correctly matched against the variance of the noise added.

Simulations were also carried out in the case of missing points. In this case, the data generated for the two frames was modified by dropping points from the first frame. It was observed that around half the points could be dropped without impairing the algorithm's ability to match the remaining points with their corresponding points in the second frame. In Table

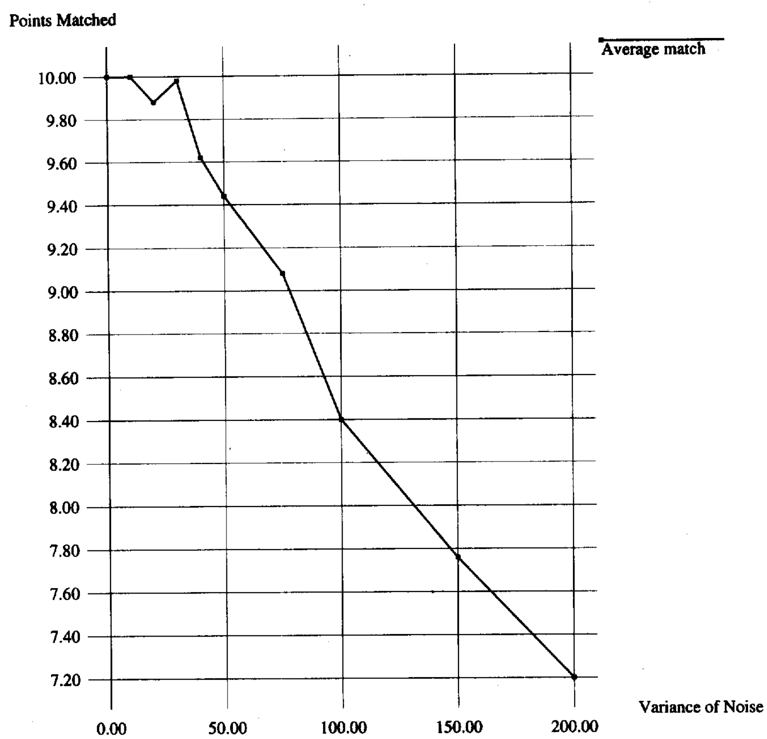


Fig. 2. Graph of average match versus variance of noise added.

IV, we show the rotation matrices computed by our method as points are successively dropped from the first frame. The frames start out with 20 points, and points are dropped, two at a time, from the first frame. Two and one points are mismatched in the cases where 12 and 14 points have been dropped, respectively. For all other cases, all points present in the first frame are correctly matched. We may note here that in this case, the noise tolerance is somewhat adversely affected.

Finally, the algorithm was applied to real objects. In the first case, we used the solids from the Shastra project [3]. These are illustrated in Figs. 3–5, and were obtained using Gati [7], the animation component of Shastra. Fig. 3 shows the initial positions of the objects and Fig. 5 their final positions when the second object was subjected to the rotation estimated by our method. Fig. 4 shows an intermediate stage. The tilt, slant, and rotation angles were 35 degrees, 45 degrees, and 40 degrees, respectively, and the actual direction cosines were (0.579227, 0.405576, 0.707110). The points used for matching were taken as the vertices of the cube of side 100, with origin as one of its corners. Our method correctly matched all points, with an average distance of 1.008 between matched points. The angles of tilt, slant, and rotation obtained from the approximation to the rotation matrix were 35.3 degrees, 45.25 degrees, and 39.672 degrees, and the direction cosines were (0.579585, 0.410445, 0.704000) The angle between the actual and obtained axis of rotation is 0.23 degrees.

Figs. 6–8 represent three frames of a cylinder being rotated about its vertical axis. Range data was obtained for this image sequence in five frames using the structured lighting system [8]. For the frames shown here, the rotation from the first to second is 10 degrees and that from second to third is 20 degrees. The first frame had five points, the second frame had all the points of the first frame and three extra points, and the third frame had all the points of the first and two extra points. The algorithm was successful in establishing correspondence between the various frames. For instance, all points from frame one were correctly matched with points in frame three. The angle of rotation was computed as 36.54 degrees, compared to the actual 30 degrees. All points were also correctly matched between frames two and three. The

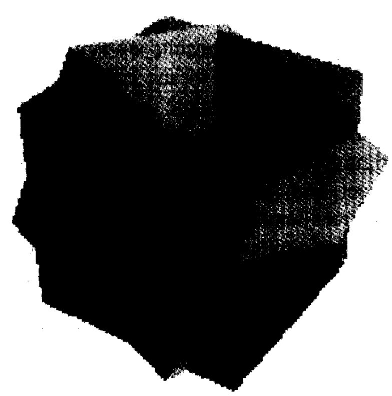


Fig. 3. Initial position of objects.

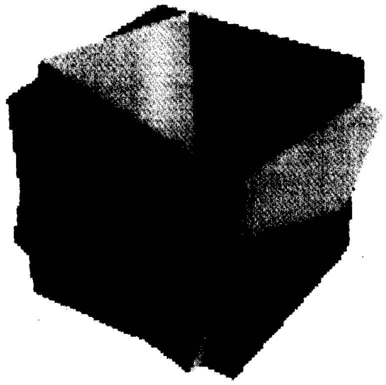


Fig. 4. Intermediate position of objects.

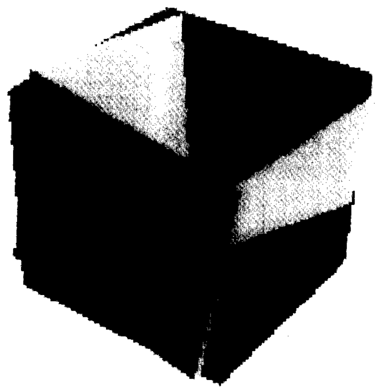


Fig. 5. Final position of objects.

angle was computed as 19.314 degrees, compared to the actual 20 degrees. The actual and computed matrices are given in Tables V and VI, respectively.

V. Discussion

In this work, we have presented a technique that utilizes concepts behind elastic nets to obtain correspondence in range data. The method uses approximations to the motion parameters to do this. Results of simulations carried out to establish the correctness of the technique have also been presented.

Huang *et al.* in [22] also use the concept of obtaining motion parameters to obtain correspondence. Their method obtains four possible rotation matrices, and the correct one is selected by a method which assumes that the exact point positions are known. Their scheme is thus very sensitive to noise. In contrast, our method is quite tolerant of noise. The scheme proposed in [21] is not as sensitive to noise, but works best if the axis of rotation is known *a priori*. Otherwise, it requires extensive computation to search for the axis of rotation before the rotation matrix can be computed. In contrast, our algorithm makes no assumption about the rotation axis being known in advance. Unlike the method proposed in [6], our method does not require any preestablished correspondences. Our method is also different from the various attribute based methods which use subgraph matching and tree pruning like approaches [11], [14], [15], and [24]. We use no attributes, solely the positions of various points. Moreover, such methods are often more useful in the case of image to model correspondence, since they often assume the existence of many views of the objects, or of an internal 3-D model of the object. In [12] and [30], motion parameters are obtained by optimization approaches. These methods are used to compute the motion parameters, however, assuming that a matching is already known.

TABLE IV

$$\mathbf{R}_{act} = \begin{bmatrix} 0.935342 & -0.306982 & 0.175778 \\ 0.340506 & 0.915986 & -0.212190 \\ -0.095872 & 0.258324 & 0.961289 \end{bmatrix}$$

Actual **R**

$$\mathbf{R}_{cmp} = \begin{bmatrix} 0.919321 & -0.298936 & 0.169291 \\ 0.338846 & 0.906692 & -0.210211 \\ -0.094456 & 0.256466 & 0.952533 \end{bmatrix}$$

Computed **R**, all points

$$\mathbf{R}_{cmp} = \begin{bmatrix} 0.901676 & -0.277623 & 0.169061 \\ 0.340636 & 0.888301 & -0.198270 \\ -0.099958 & 0.265297 & 0.947453 \end{bmatrix}$$

2 missing points

$$\mathbf{R}_{cmp} = \begin{bmatrix} 0.886497 & -0.280634 & 0.155107 \\ 0.324547 & 0.892247 & -0.207362 \\ -0.091173 & 0.263534 & 0.954792 \end{bmatrix}$$

4 missing points

$$\mathbf{R}_{cmp} = \begin{bmatrix} 0.833107 & -0.285937 & 0.126678 \\ 0.285545 & 0.885076 & -0.226767 \\ -0.124114 & 0.263495 & 0.948059 \end{bmatrix}$$

6 missing points

$$\mathbf{R}_{cmp} = \begin{bmatrix} 0.877222 & -0.274965 & 0.163671 \\ 0.301456 & 0.895053 & -0.199683 \\ -0.095423 & 0.282237 & 0.917954 \end{bmatrix}$$

8 missing points

$$\mathbf{R}_{cmp} = \begin{bmatrix} 0.914605 & -0.184780 & 0.116181 \\ 0.311168 & 0.877728 & -0.201504 \\ -0.111929 & 0.258657 & 0.970692 \end{bmatrix}$$

10 missing points

$$\mathbf{R}_{cmp} = \begin{bmatrix} 0.741863 & -0.133472 & -0.009451 \\ 0.242434 & 0.849438 & -0.214434 \\ -0.184199 & 0.249792 & 0.935028 \end{bmatrix}$$

12 missing points

$$\mathbf{R}_{cmp} = \begin{bmatrix} 0.642597 & -0.046344 & -0.028805 \\ 0.240439 & 0.903996 & -0.291303 \\ -0.038989 & 0.181455 & 0.979305 \end{bmatrix}$$

14 missing points

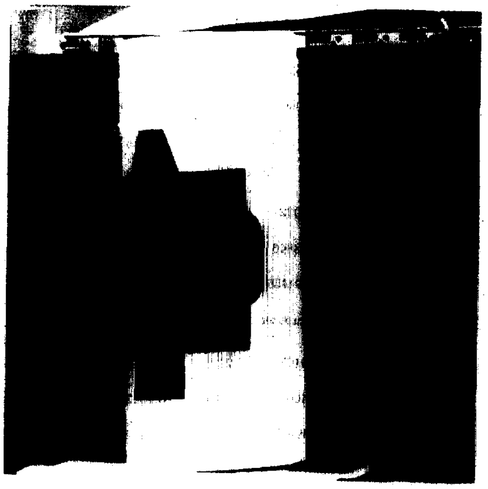


Fig. 6. First frame of rotating cylinder.

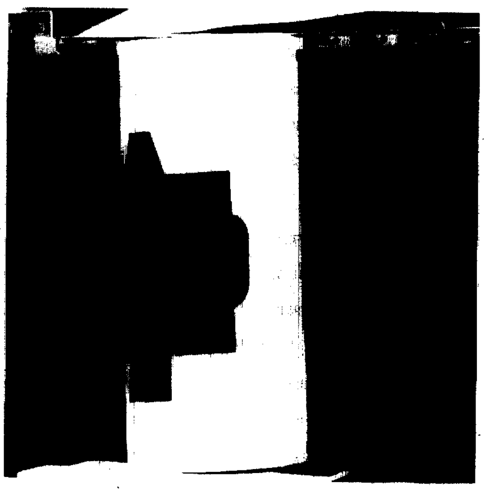


Fig. 7. Second frame of rotating cylinder.

Unlike most of the methods in the literature, our algorithm also works very well in cases where not all points of one of the frames are found in the other. We have shown results in the case where one of the frames has fewer points than the other, but all the points in the smaller frame are found in the other. This condition would hold very well for image to model kind of matches, but in a dynamic case some frame sequences may not satisfy this constraint. In that case, two approaches are possible.

- Since we make no constraints on the extent of motion, a frame can always be compared with some other preceding frame such that our condition is met. If a frame, say m, has all the points of frames i and $i+1$ neither of which contains all the points of the other, we can establish correspondence between frames i and $i+1$ by establishing correspondences between frames i and m and $i+1$ and m. Thus we see that our assumption is not very restricting.

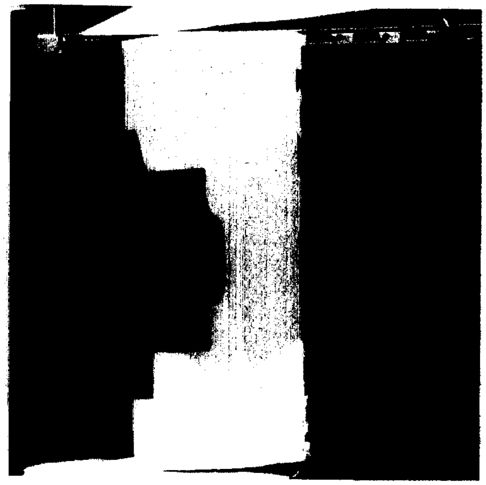


Fig. 8. Third frame of rotating cylinder.

TABLE V
ACTUAL AND COMPUTED r FOR CYLINDER FRAMES 2 AND 3.

$$\mathbf{R}_{act} = \begin{bmatrix} 0.939693 & 0 & 0.342018 \\ 0 & 1 & 0 \\ -0.342018 & 0 & 0.939693 \end{bmatrix}$$

$$\mathbf{R}_{cmp} = \begin{bmatrix} 0.940022 & 0.073742 & 0.352072 \\ -0.004055 & 1.012843 & 0.006679 \\ 0.021357 & -0.124675 & 0.934572 \end{bmatrix}$$

TABLE VI
ACTUAL AND COMPUTED R FOR CYLINDER FRAMES 1 AND 3.

$$\mathbf{R}_{act} = \begin{bmatrix} 0.866027 & 0 & 0.499998 \\ 0 & 1 & 0 \\ -0.499998 & 0 & 0.866027 \end{bmatrix}$$

$$\mathbf{R}_{cmp} = \begin{bmatrix} 0.755679 & -0.049039 & 0.084014 \\ 0.117516 & 1.003520 & 0.000984 \\ -0.383027 & 0.058693 & 0.847536 \end{bmatrix}$$

- If, in a pathalogical case, there be no such frame m, we can compute the average distance between the matched pairs. The pairings whose distance is more than k times this average can be rejected as being invalid matches, k being some suitably chosen constant greater than one.

The ability of our algorithm to work despite missing points and in presence of noise makes it suitable to work on real range sequences.

ACKNOWLEDGMENT

The authors would like to thank the anonymous referees for their comments and suggestions. The authors would also like to thank S. Cutchin, V. Anupam, and C. Bajaj, of Project

Shastra at Purdue, and E. Fisher and Stanley Dunn, of the Biomedical Engineering Department of Rutgers University, for their help in obtaining and rendering some of the range data. Thanks are also due to J. Peters for his help with some aspects of numerical analysis and W. Gorman for his help with the presentation.

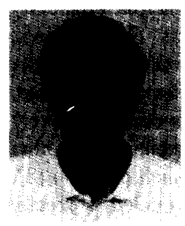
REFERENCES

[1] J. K. Aggarwal, "Motion and time varying imagery—an overview," in *Proc. IEEE Workshop Motion: Representation Anal.*, 1986, pp. 1–6.
[2] J. Aloimonos, "Perception of structure from motion," in *Proc. Conf. Comput. Vision Pattern Recog.*, 1986.
[3] V. Anupam and C. L. Bajaj, "SHASTRA–An architecture for development of collaborative applications," *Int. J. Intell. Coperative Info. Syst., vol. 3, no. 2, pp. 155–172, 1994.*.
[4] H. Baker, "Depth from edge and intensity based stereo," Ph.D. dissertation, Dep. Comp. Sci., Stanford Univ., 1981.
[5] W. Burger and B. Bhanu, "Estimating 3-D motion from perspective image sequences," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 12, no. 11, pp. 1040–1058, 1990.
[6] Homer H. Chen and T. S. Huang, "Maximal matching of two 3-D point sets," in *Proc. Int. Conf. Pattern Recog.*, 1986, pp. 1048–1050.
[7] S. Cutchin and C. L. Bajaj, "The gati client/server animation toolkit," in *Proc. CGI*, 1993, pp. 413–423.
[8] S. M. Dunn, R. L. Keizer, and J. D. Yu, "Measuring the area and volume of the human body with structured light," *IEEE Trans. Syst., Man Cyber.*, vol. 19, pp. 1350–1364, 1989.
[9] R. Durbin, R. Szeliski, and A. L. Yuille, "An analysis of the elastic net approach to the travelling salesman problem," *Neural Computation*, vol. 1, pp. 348–358, 1989.
[10] R. Durbin and D. Willshaw, "An analogue approach to the travelling salesman problem using an elastic net method," *Nature*, vol. 326, pp. 689–691, 1987.
[11] T. J. Fan, G. Medioni, and R. Nevatia, "Recognizing 3D objects using surface descriptions," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 11, pp. 1140–1157, 1989.
[12] O. D. Faugeras and M. Herbert, "A 3-D recognition and positioning algorithm using geometrical matching between primitive surfaces," *IJCAI*, pp. 996–1002, 1983.
[13] O. D. Faugeras and S. Maybank, "Motion from point matches: Multiplicity of solutions," in *Proc. IEEE Workshop Motion*, 1989.
[14] P. J. Flynn and A. K. Jain, "Bonsai, a 3D object recognition using constrained search," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 13, no. 10, pp. 1066–1075, 1991.
[15] W. E. L. Grimson and T. Lozano-Perez, "Model based recognition and localization from sparse range or tactile data," *Int. Robotics Res. J.*, vol. 3, no. 3, pp. 3–35, 1984.
[16] B. K. P. Horn, "Recovering baseline and orientation from essential matrix," MIT AI Memo, Tech. Rep., 1990.
[17] ——, "Relative orientation," *Int. J. Comput. Vision*, vol. 4, pp. 59–78, 1990.
[18] R. Jain and I. K. Sethi, "Finding trajectories of feature points in a monocular image sequence," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 9, pp. 56–73, 1987.
[19] Y. G. Leclerc and S. W. Zucker, "The local structure of image discontinuities in one dimension," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 9, pp. 341–355, 1987.
[20] C. H. Lee and A. Joshi, "Correspondence problem in image sequence analysis," *Pattern Recognition*, vol. 26, no. 1, pp. 47–61, 1993.
[21] Z. C. Lin and T. S. Huang *et al.*, "Motion estimation from 3D point sets with and without correspondence," in *Proc. Conf. Comput. Vision Pattern Recognition*, 1986, pp. 194–201.
[22] Z. C. Lin, H. Lee and T. S. Huang, "Finding 3D point correspondences in motion estimation," in *Proc. Conf. Pattern Recognition*, 1986, pp. 303–305.
[23] H. C. Longuet-Higgens, "A computer algorithm for reconstructing a scene from two projections," *Nature*, vol. 293, pp. 133–135, 1981.
[24] M. J. Magee, B. A. Boyter, C. H. Chien and J. K. Aggarwal, "Experiments in intensity guided range sensing recognition of three dimensional objects," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 7, pp. 629–636, 1985.
[25] H. H. Nagel, "Image sequences—Ten (octal) years—From phenomenology toward a theoretical foundation," in *Proc. Int. Conf. Pattern Recognition*, 1986, pp. 1174–1185.
[26] N. M. Nasrabadi and C. Y. Choo, "Hopfield network for stereo correspondence," *IEEE Trans. Neural Networks*, vol. 3, no. 1, pp. 5–13, 1992.
[27] K. Rangarajan and M. Shah, "Estimating motion correspondence," *Proc. Conf. Comput. Vision Pattern Recog.*, 1991.
[28] J. W. Roach and J. K. Aggarwal, "Determining the movement of objects from a sequence of images," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 2, pp. 29–46, 1979.
[29] V. Salari and I. K. Sethi, "Feature point correspondence in presence of occlusion," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 12, pp. 87–91, 1990.
[30] M. D. Shuster, "Approximate algorithms for fast optimal attitude computation," in *Proc. AIAA Guidance Contr. Specialists Conf.*, 1978, pp. 88–95.
[31] R. Y. Tsai and T. S Huang, "Uniqueness and estimation of three dimensional motion parameters of rigid objects with curved surfaces," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 6, no. 1, pp. 13–27, 1984.
[32] S. Ullman, *The Interpretation of Visual Motion*. Cambridge, MA: MIT Press, 1979.
[33] J. Weng *et al.*, "3D motion estimation, understanding and prediction from noisy image sequences," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 9, pp. 370–389, 1987.
[34] T. D. Williams, "Depth from camera motion in a real world scene," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 2, pp. 511–516, 1980.
[35] G. S. Young and Rama Chellappa, "3-D motion estimation using a sequence of noisy images," in *Proc. Conf. Comput. Vision Pattern Recognition*, 1988.
[36] A. L. Yuille, "Generalized deformable models, statistical physics, and matching problems," *Neural Computation*, vol. 2, pp. 1–24, 1990.
[37] D. Zhang, "Perspective invariant description of a planar point set and its application to matching," in *Proc. Int. Conf. Pattern Recog.*, 1986.

Anupam Joshi (M'93) was born in New Delhi, India. He received the Bachelor of Technology degree in electrical engineering from the Indian Institute of Technology, Delhi, in 1989, and the Ph.D. degree in computer science from Purdue University in 1993.

He is currently a Visiting Assistant Professor in the Department of Computer Sciences at Purdue. His research interests are in the areas of computational vision, artificial neural systems, high performance/scientific computing, mobile computing and related hardware topics.

Dr. Joshi is a member of ΥΠΕ and ACM.



Chia-Hoang Lee received the Ph.D. degree in computer science from the University of Maryland, College Park, in 1983.

Dr. Lee is currently the chair of the Department of Computer and Information Science, National Chiao-Tung University, Hsinchu, Taiwan, R.O.C. From 1984–1985, he was a faculty member of the Department of Computer Science and Mathematics at the University of Maryland, Baltimore County. From 1985–1992, he was a faculty member in the Computer Science Department of Purdue University. His current research interests include computational vision, neural networks and multimedia.