New Algorithms for 2D and 3D Point Matching: Pose Estimation and Correspondence

Steven Gold
CuraGen Corporation
322 East Main Street
Branford, CT 06405
e-mail: gold@curagen.com

Anand Rangarajan
Dept. of Diagnostic Radiology
Yale University School of Medicine
New Haven, CT 06520-8042
e-mail: anand@noodle.med.yale.edu

Chien-Ping Lu
Silicon Graphics Inc.
2011 North Shoreline Blvd.
Mountain View, CA 94039
e-mail: cplu@engr.sgi.com

Suguna Pappu
Dept. of Diagnostic Radiology
Yale University School of Medicine
New Haven, CT 06520-8042
e-mail: pappu@noodle.med.yale.edu

Eric Mjolsness
Department of Computer Science and Engineering
University of California San Diego (UCSD)
La Jolla, CA 92093-0114
e-mail: emj@cs.ucsd.edu

January 3, 1997

Abstract

A fundamental open problem in computer vision—determining pose and correspondence between two sets of points in space—is solved with a novel, fast, robust and easily implementable algorithm. The technique works on noisy 2D or 3D point sets that may be of unequal sizes and may differ by non-rigid transformations. Using a combination of optimization techniques such as deterministic annealing and the *softassign*, which have recently emerged out of the recurrent neural network/statistical physics framework, analog objective functions describing the problems are minimized. Over thirty thousand experiments, on randomly generated points sets with varying amounts of noise and missing and spurious points, and on hand-written character sets demonstrate the robustness of the algorithm.

Keywords: Point-matching, pose estimation, correspondence, neural networks, optimization, softassign, deterministic annealing, affine.

1 Introduction

Matching the representations of two images has long been the focus of much research in Computer Vision, forming an essential component of many machine-based object recognition systems. Critical to most matching techniques is the determination of correspondence between spatially localized features within each image. Often such features are treated as points in 2 or 3 dimensional space. The resulting point matching problem is difficult to solve—especially when issues of noise, missing or spurious data, and non-rigid transformations are tackled (Grimson, 1990).

Using a new technique—the *softassign*—which has emerged from the recurrent neural network/statistical physics framework, we develop new algorithms for 2D and 3D point matching. Energy functions are formulated for 2D and 3D point matching problems. These energy functions are characterized by the use of a match matrix to explicitly denote an assignment (correspondence) between one set of points and another. The match matrix is a zero-one matrix with one's denoting that a point in one set is assigned to (corresponds to) a point in the other set. The match matrix variables must satisfy *assignment matrix* constraints, i.e. the rows and columns must add up to one (except for the row and column holding the slack variables) and the entries must be zero or one. In the case of a square matrix (where the sets to be matched are equal in size) the match matrix is a permutation matrix. Assignment matrix constraints may also be described as two-way interlocking winner-take-all (WTA) constraints.

The original Hopfield-Tank neural network (Hopfield and Tank, 1985) had great difficulty in satisfying assignment matrix constraints since it used *penalty functions* with fixed parameters. As a result the quality of the solutions obtained for many optimization problems, like the traveling salesman problem, was poor (Wilson and Pawley, 1988; Kamgar-Parsi and Kamgar-Parsi, 1990). More recently, a number of techniques from statistical physics have been adopted to mitigate these problems. These include deterministic annealing which convexifies the energy function in order to avoid some local minima and the Potts glass approximation which results in a hard enforcement of one-way (one set of) winner-take-all constraints (WTA) via the softmax (Peterson and Soderberg, 1989; Van den Bout and Miller III, 1990; Simic, 1991).

However, when the problem calls for assignment constraints, as does point matching, the resulting energy function must still include a penalty term when the softmax is employed in order to enforce the second set of WTA constraints. Such penalty terms may introduce spurious local minima in the energy function and involve free parameters which are hard to set. A new technique termed *softassign* eliminates the need for all such penalty terms and their associated free parameters. The first use of the softassign was in an algorithm for the assignment problem (Kosowsky and Yuille, 1994). It has since been applied to much more difficult optimization problems, including many special cases of quadratic assignment problems such as graph matching, TSP, and graph partitioning (Gold et al., 1996; Rangarajan et al., 1996; Gold and Rangarajan, 1996a; Gold, 1995; Gold and Rangarajan, 1996b). Here for the first time it is applied to point matching, which is formulated as a parametric assignment problem. By formulating it in this manner we are able to combine the estimation of both pose and correspondence by minimizing a single non-linear objective function.

The result is a new, fast, robust and easily implementable algorithm to find the pose and correspondence between noisy 2D or 3D unlabeled point sets despite missing or spurious points. It is derived by minimizing an analog global objective function using a combination of optimization techniques incorporating deterministic annealing and the softassign. These new optimization methods (Rangarajan et al., 1996; Gold and Rangarajan, 1996b; Gold, 1995), not previously applied to point-matching, result in an accurate and fast algorithm in the presence of substantial noise and a high percentage of missing or spurious features. Over thirty thousand experiments, on randomly generated points sets with varying amounts of noise and missing and spurious points, as well as on hand-written character sets demonstrate the robustness of the algorithm.

2 Related Work

A large number of different approaches have been tried on point matching. Tree-pruning methods involve searching over a tree of possible matches while eliminating portions of the search space (Baird, 1984; Grimson and Lozano-Perez, 1987; Umeyama, 1993). The generalized Hough transform requires the division of the parameter space of possible poses into discrete bins wherein good matches are registered as votes in the appropriate bin (Ballard, 1981; Stockman, 1987). Geometric hashing is another voting scheme where discrete bins are created for the possible bases that can be used to represent the point sets (Lamdan et al., 1988; Hummel and Wolfson, 1988). In the alignment method (Ullman, 1989) each alignment feature (defined as a set of three distinctive points) in the image is matched against each alignment feature in the model, from which a pose is obtained. Subsequently the best such pose is chosen. More recently, probabilistic techniques have been applied to enhance the speed of this alignment method (Olson, 1995). Considerable attention has focused on the computation of invariants which characterize small groups of point features despite unknown geometric transformations (e.g. rigid Euclidean, affine, perspective, and so on) (Weinshall and Tomasi, 1995; Jacobs, 1994) and which can then be used in matching and indexing algorithms, such as geometric hashing (Califano and Mohan, 1994; Lamdan et al., 1988). Efficient matching algorithms between point sets have resulted from minimizing their Hausdorff distance (Huttenlocher et al., 1993), which can be made robust against missing points but which does not appear to have a statistical noise model underlying it since one point in one set can be the closest point to many points in the other set. The method is used on edge images with many points but little independent point jitter. A related distance measure and matching method in (Besl and McKay, 1992) also uses closest points.

Techniques more closely related to our neural network algorithms are eigenvector based approaches, relaxation labeling algorithms, deformable object modeling and other neural network methods (there is a large overlap in methods and ideas within these four categories). (Scott and Longuet-Higgins, 1991) have a formulation for point matching that is similar to ours, with a pairing matrix that indicates matches and a proximity matrix whose elements are the pairwise distance between points. The proximity matrix is a function of a Gaussian weighted distance metric, with a parameter σ which controls the degree of interaction between the two sets of features. Their eigenvector based approach computes the modes of the proximity matrix, and they show that for a value of σ large enough, they recover the correct global correspondence. (Shapiro and Brady, 1992) continue this work by including modal shape information to address the weaknesses of (Scott and Longuet-Higgins, 1991)'s algorithm, primarily its inability to recover large rotations in the transformations, and also that the assumption of large σ may result in algorithmic instabilities.

Deformable object models can be used to define distance measures between point feature sets, based on the eigenmodes of an underlying unsampled object model (Sclaroff and Pentland, 1995). These may then be used to solve for correspondence. Deformable template models are elaborated in (Yuille, 1990; Yuille and Hallinan, 1992) and elsewhere; they provide a physically-based nonlinear optimization approach to matching but must be separately related to sparse image data such as point sets, or related directly to images instead. For example elastic "snake" models (Kass et al., 1988) are now commonly used to represent unknown curves and to pick them out of intensity or gradient images. Relating models to image intensities rather than to feature points is favored in face recognition experiments of (Brunelli and Poggio, 1993) and in the more neural network like approach of (Hinton et al., 1992).

Relaxation labeling algorithms first introduced by (Ranade and Rosenfeld, 1980) have also been widely applied to point matching. However these methods were originally developed as tools for classification and consequently in general only impose one-way constraints and not the two-way constraints required for many point matching problems. That is, there is a constraint that a point in one set can match to only one point in the other set, but there is no similar constraint for the points in the second set, i.e. there is no two-way WTA (assignment) constraint. (Ton and Jain, 1989) attempt to impose such a two-way constraint within the relaxation labeling framework but do not use other key techniques such as deterministic annealing, incorporated in algorithms described in this paper. (Li, 1992) use a form of deterministic annealing (graduated non-convexity) within the relaxation labeling framework, but only with a one-way constraint.

More recently, within the neural network community, several researchers have attempted to formulate and solve the point matching problem by minimizing an objective function which handles both pose and correspondence. In a series of papers, (Mjolsness and Garrett, 1990; Mjolsness, 1991; Lu and Mjolsness, 1994), several closely related objective functions containing correspondence and pose parameters were minimized. None of these methods explicitly handled two-way constraints. Similarly, in (Hinton et al., 1992), an objective function with affine and correspondence parameters was minimized but only a one-way constraint was imposed. Similar problems related to two-way constraint satisfaction beset the objective function based methods of (Vinod and Ghose, 1993; Gee et al., 1993).

3 2D with Affine Transformations

3.1 Formulating the Objective

In our first point matching problem, we assume we are given two 2D point sets, $\{X_j\}$ and $\{Y_k\}$, related by an affine transformation $\{A, t\}$. We assume the affine transform is bounded in size—for example order of magnitude differences in scale between the point sets is not permitted. The positions of each point in 2D space are noisy; they may be considered as arising from Gaussian distributions, whose means correspond to the exact x-y coordinates in the absence of noise. Points can be deleted from or added to each point set, i.e. outliers may be present in each set. We then define a set of correspondence variables $\{m_{jk}\}$—our *match matrix*—such that:

$$m_{jk} = \begin{cases} 1 & \text{if point } X_j \text{ corresponds to point } Y_k \\ 0 & \text{otherwise,} \end{cases}$$

Now our problem may be defined as: Given two such sets of points $\{X_j\}$ and $\{Y_k\}$ find the affine transformation $\{A, t\}$—the *pose*—and the match matrix $\{m_{jk}\}$—the *correspondence*—that best relates them. Often a point matching algorithm will attempt to find either the pose or the correspondence, since knowledge of one relatively easily determines the other; i.e. given the pose one can determine the correspondence or given the correspondence one can determine the pose. However, in our algorithm we will determine both simultaneously, producing estimates for first the correspondence and then the pose and going back and forth between the correspondence and pose in an iterative manner. This two stage iterative algorithm will naturally arise from the minimization of an energy function describing the problem, using techniques that have evolved out of the neural network/statistical physics framework.

Therefore given two sets of points $\{X_j\}$ and $\{Y_k\}$ we formulate the following objective to find the affine transformation, $\{A, t\}$, and correspondence or match matrix, $\{m_{jk}\}$, which best maps some points of X onto some points of Y:

$$E_{2D}(m, t, A) = \sum_{j=1}^{J} \sum_{k=1}^{K} m_{jk} \|X_j - t - AY_k\|^2 + g(A) - \alpha \sum_{j=1}^{J} \sum_{k=1}^{K} m_{jk} \tag{1}$$

subject to $\forall j \ \sum_{k=1}^{K} m_{jk} \leq 1$, $\forall k \ \sum_{j=1}^{J} m_{jk} \leq 1$, $\forall jk \ m_{jk} \in \{0, 1\}$ and

$$g(A) = \gamma(a^2 + b^2 + c^2) \ .$$

A (composed of four separate parameters $\{a, \Theta, b, c\}$) is decomposed into scale, rotation, and two components of shear as follows:

$$A = s(a)R(\Theta)Sh_1(b)Sh_2(c)$$

where,

$$s(a) = \begin{pmatrix} e^a & 0 \\ 0 & e^a \end{pmatrix} , \ Sh_1(b) = \begin{pmatrix} e^b & 0 \\ 0 & e^{-b} \end{pmatrix} , \ Sh_2(c) = \begin{pmatrix} \cosh(c) & \sinh(c) \\ \sinh(c) & \cosh(c) \end{pmatrix}$$

$R(\Theta)$ is the standard 2x2 rotation matrix. $g(A)$ serves to regularize (Girosi et al., 1995) the affine transformation by penalizing large values of the scale and shear components. Three separate regularization parameters, γ, κ and λ can also be used for this purpose. If different bounds are desired for the different components of A we could set $g(A) = \gamma a^2 + \kappa b^2 + \lambda c^2$. However in all the experiments in this paper, only one parameter γ, was used to set the bound on the scale and shear components. The constraints on our match matrix, $\{m_{jk}\}$, ensure that each point in each image corresponds to at most one point in the other image. The inequality constraints on $\{m_{jk}\}$ permit null matches, i.e. permit outliers.

The α term biases the objective towards matches. It acts as a threshold error distance, indicating how far apart two points must be (or how much noise our system can tolerate) before the points must be treated as outliers. If for any pair of points $\{X_{\hat{j}}, Y_{\hat{k}}\}$ and a current estimate of $\{A, t\}$, $\|X_{\hat{j}} - t - AY_{\hat{k}}\|^2 < \alpha$ then $X_{\hat{j}}$ will not be considered an outlier with respect to $Y_{\hat{k}}$ and vice versa since the objective will now favor (be lower in value) $m_{\hat{j},\hat{k}} = 1$ over $m_{\hat{j},\hat{k}} = 0$ *assuming all other m's in the jth row and kth column are zero.*

The decomposition of A in the above is not required, since A could be left as a 2x2 matrix and solved for directly in the algorithm that follows (Pappu et al., 1996). The decomposition just provides for more precise regularization, i.e., specification of the likely kinds of transformations. So, for example, in the experiments in Section 5.2 we ran one set of experiments using the full affine transformation and another set using just the scale, rotation and translation components of the affine. Also $Sh_2(c)$ could be replaced by another rotation matrix, using the singular value decomposition of A.

In the above objective there are only two parameters, γ and α that need to be adjusted—both of which are dependent upon the problem domain. While for the experiments in this paper we simply set these two parameters—thereby also fixing the distribution of the population assumed in those experiments, it is possible to use a maximum likelihood method to estimate these parameters based on a sample population if our distribution were unknown (Duda and Hart, 1973).

3.2 The Softassign—An Intuitive Development

The major hurdle in finding good suboptimal solutions to the point matching objective (1) is satisfying the two-way constraints, i.e. the row and column constraints on the match matrix (Figure 1) together with the constraint that the individual entries of m be zero or one.

We will ignore the inequality constraints (ignore for the moment the slacks in Figure 1) on

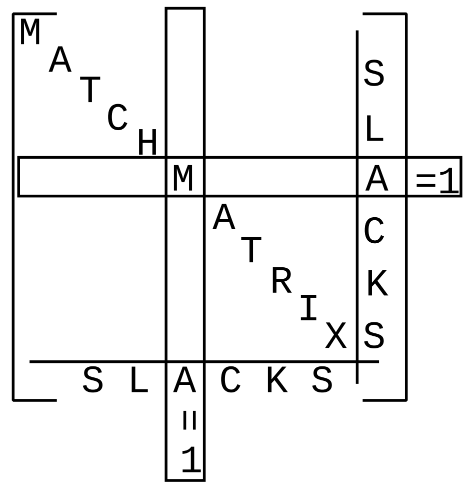


Figure 1: The match matrix, m

the rows and columns, in order to simplify the following development. Therefore, the constraints state that our match matrix must be a permutation matrix. (A permutation matrix is a square zero-one matrix whose rows and columns add up to one.) We now use *deterministic annealing* methods (Peterson and Soderberg, 1989; Geiger and Girosi, 1991) to turn our discrete problem into a continuous one in order to reduce the chances of getting trapped in local minima. This method consists of minimizing a series of objective functions indexed by a control parameter. As the parameter is increased the solution to the objective functions approach that of the discrete problem. The major problem now is the minimization of the point matching objective subject to the usual two-way constraints on the match matrix and the new constraint that the individual entries of m lie in the interval $[0, 1]$. The constraints are relaxed from permutation matrix constraints to *doubly stochastic matrix* constraints. A doubly stochastic matrix is a square matrix with all positive entries and rows and columns summing to one—it may roughly be thought of as the continuous analog of a permutation matrix.

First, we will examine the case where there is only one constraint. Imagine a subproblem (within a larger problem) whose objective is to find the maximum element within a set of numbers (WTA). That is, we are given a set of variables $\{Q_j\}$ where $Q_j \in R^1$. Then, we associate a variable $m_j \in \{0, 1\}$ with each Q_j, such that $\sum_{j=1}^{J} m_j = 1$. Our aim is:

$$m_{\hat{j}} = \begin{cases} 1 & \text{if } Q_{\hat{j}} \text{ is the maximum number in } \{Q_j\} \\ 0 & \text{otherwise,} \end{cases}$$

which is equivalent to finding $\{m_j\}$ which maximize $\sum_{j=1}^{J} m_j Q_j$. This discrete problem may now be formulated as a continuous problem by introducing a control parameter $\beta > 0$ and then setting m as follows (Peterson and Soderberg, 1989; Geiger and Yuille, 1991) :

$$m_{\hat{j}} = \frac{\exp(\beta Q_{\hat{j}})}{\sum_{j=1}^{J} \exp(\beta Q_j)}$$

This is known as the *softmax* (Bridle, 1990). The exponentiation used within softmax has the effect of ensuring that all the elements of $\{m_j\}$ are positive. It is easily shown that as β is increased in the above, the m_j corresponding to the maximum Q_j approaches 1 while all the other m_j approach 0 (except in special cases of ties). In the limit as $\beta \to \infty$, the m_j corresponding to the maximum will equal 1 while all the other m_j will equal 0. Therefore an algorithm using a deterministic annealing method to enforce a constraint which selects the maximum among a group of elements could have the following form:

Initialize β to β_0

Begin A: (Do A until $(\beta \geq \beta_f)$)

$m_j^0 \leftarrow \exp(\beta Q_j)$

$m_j^1 \leftarrow \frac{m_j^0}{\sum_{j=1}^{J} m_j^0}$

Do rest of algorithm - (Q_j's may be updated) . . . .

increase β

End A

However, in our problem we have two-way WTA constraints: A point in set X must correspond to only one point in set Y and vice versa. With the adoption of deterministic annealing, m_{jk} can assume values inside the unit hypercube but still has to satisfy doubly stochastic matrix constraints. Fortunately, doubly stochastic constraints can be satisfied using a remarkable result due to Sinkhorn (Sinkhorn, 1964). Sinkhorn (Sinkhorn, 1964) proves that a doubly stochastic matrix is obtained from any square matrix with all positive entries by the iterative process of alternating row and column normalizations. Imagine a subproblem (within a larger problem) whose objective is to find the best (maximum) assignment given a square benefit matrix of numbers. That is, we are given a set of variables $\{Q_{jk}\}$ where $Q_{jk} \in R^1$. Then we associate a variable $m_{jk} \in \{0, 1\}$ with each Q_{jk}, such that $\forall j \ \sum_{k=1}^{K} m_{jk} = 1$ and $\forall k \ \sum_{j=1}^{J} m_{jk} = 1$. Our aim is to find the matrix m (a permutation matrix) which maximizes the following:

$$E_a(m) = \sum_{j=1}^{J} \sum_{k=1}^{K} m_{jk} Q_{jk}$$

This is known as the assignment problem, a classic problem in combinatorial optimization

(Papadimitriou and Steiglitz, 1982). Therefore an algorithm using a deterministic annealing method to enforce a two-way constraint which selects the maximum assignment among a group of elements could have the following form:

Initialize β to β_0

Begin A: (Deterministic annealing) (Do A until ($\beta \geq \beta_f$))

$m^0_{jk} \leftarrow \exp(\beta Q_{jk})$

Begin B: (Sinkhorn's method) (Do B until m converges)

Update m by normalizing across all rows:

$m^1_{jk} \leftarrow \frac{m^0_{jk}}{\sum_{k=1}^{K} m^0_{jk}}$

Update m by normalizing across all columns:

$m^0_{jk} \leftarrow \frac{m^1_{jk}}{\sum_{j=1}^{J} m^1_{jk}}$

End B

Do rest of algorithm - (Q_{jk}'s may be updated) . . . .

increase β

End A

Note that the exponentiation used has the effect of ensuring that all the elements of the match matrix are positive before Sinkhorn's method is applied. Just such an algorithm was used in (Kosowsky and Yuille, 1994) to exactly solve the assignment problem (the global maximum is found). However, the point matching problem we are trying to solve is much harder than the linear assignment problem which can be solved in polynomial time (Bertsekas and Tsitsiklis, 1989). Since we have already described a method to solve the assignment problem, we will find an approximate solution to our parametric assignment problem by using a deterministic annealing method to solve a succession of assignment problems. For each assignment the method returns the corresponding globally optimal doubly stochastic matrix for the current value of the control parameter (Kosowsky and Yuille, 1994). Since a doubly stochastic matrix (and not a permutation matrix) is returned for each assignment problem at the current value of the control parameter we term this a softassign.

Recall our point matching problem corresponds to the minimization of the objective (rearranging terms) $\sum_{j=1}^{J}\sum_{k=1}^{K} m_{jk}(\|X_j - t - AY_k\|^2 - \alpha) + g(A)$ subject to assignment constraints. For fixed $\{A, t\}$ it is easy to see this is just an assignment problem where $Q_{jk} = -(\|X_j - t - AY_k\|^2 - \alpha) = -\frac{\partial E_{2D}}{\partial m_{jk}}$. (The sign is reversed because we are minimizing, instead of maximizing as in the canonical form of the assignment problem.) We estimate the values of $\{A, t\}$ (our pose parameters), subsequently used for each new assignment problem by doing one step of coordinate descent on those parameters (resulting in an update to our $\{Q_{jk}\}$ assignment matrix as in the immediately preceding pseudocode above). See Figure 2 for an overview of the resulting point matching algorithm. Figure 3 details the softassign.

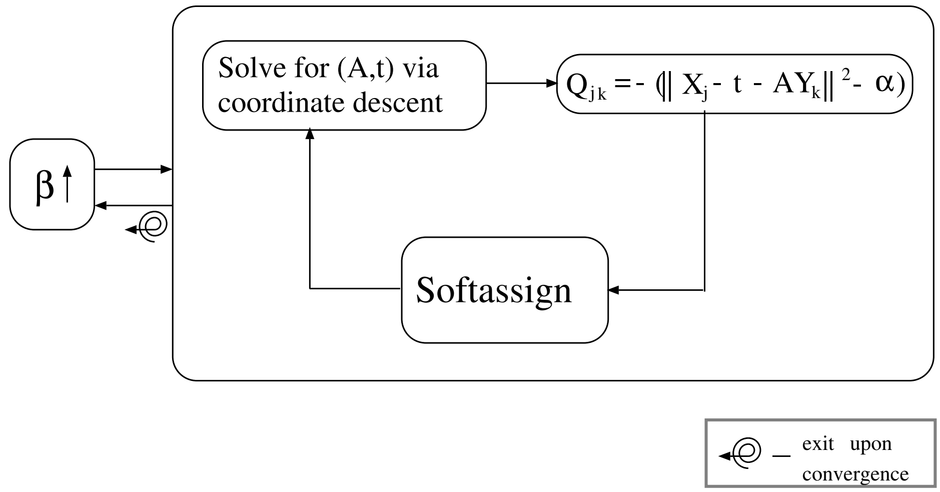


Figure 2: Overview of the point matching algorithm. $\{A, t\}$ are the affine parameters updated in step E in the pseudocode of the algorithm in the text and $\{m_{jk}\}$ is the match matrix.

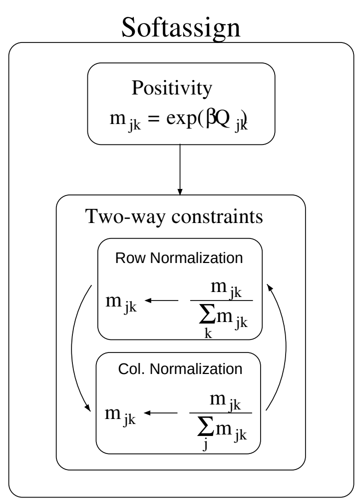


Figure 3: The softassign.

The result is a simple two step iterative algorithm. In step 1 we estimate our correspondence parameters (the match matrix) using the softassign. In the step 2 we estimate our pose parameters using coordinate descent.

One last detail needs to be resolved. The constraints on m are inequality constraints, not equality constraints. Our algorithm must handle unequal point sets! Therefore, we transform the inequality constraints into equality constraints by introducing slack variables, a standard technique from linear programming (Chvatal, 1983);

$$\forall j \; \sum_{k=1}^{K} m_{jk} \leq 1 \;\; \rightarrow \;\; \forall j \; \sum_{k=1}^{K+1} m_{jk} = 1$$

and likewise for our column constraints. An extra row and column are added to the matrix m to hold the slack variables (Figure 1). (This augmented matrix is denoted by $\hat{m}$). By incorporating slack variables, the point matching algorithm can handle outliers (spurious or missing points) in a statistically robust manner (Black and Rangarajan, 1996).

3.3 Pseudocode for the Algorithm

The pseudocode for the point matching algorithm is as follows (using the variables and constants defined below):

Initialize Θ, t, a, b, and c to zero, β to β_0, $\hat{m}_{jk}$ to $(1+\epsilon)$, γ to γ_0

Begin A: Do A until $(\beta \geq \beta_f)$

  Begin B: Do B until m converges or # of iterations $> I_0$

    Begin C (update correspondence parameters by softassign):

      $Q_{jk} \leftarrow -\frac{\partial E_{2D}}{\partial m_{jk}}$

      $m^0_{jk} \leftarrow \exp(\beta Q_{jk})$

      Begin D: Do D until $\hat{m}$ converges or # of iterations $> I_1$

        Update $\hat{m}$ by normalizing across all rows:

        $\hat{m}^1_{jk} \leftarrow \frac{\hat{m}^0_{jk}}{\sum_{k=1}^{K+1} \hat{m}^0_{jk}}$

        Update $\hat{m}$ by normalizing across all columns:

        $\hat{m}^0_{jk} \leftarrow \frac{\hat{m}^1_{jk}}{\sum_{j=1}^{J+1} \hat{m}^1_{jk}}$

      End D

    End C

    Begin E (update pose parameters by coordinate descent):

      Update Θ using analytical solution

      Update t using analytical solution

      Update a using Newton's method

      Update b using Newton's method

      Update c using Newton's method

    End E

End B

$\beta \leftarrow \beta_r \beta$, $\gamma \leftarrow \gamma / \beta_r$

End A

Variable and constant definitions can be found in Table 1.

β	control parameter of the deterministic annealing method
β_0	initial value of the control parameter β
β_f	maximum value of the control parameter β
β_r	rate at which the control parameter β is increased
γ	regularization parameter for scale and shear components of affine
γ_0	initial value of the regularization parameter γ
E_{2D}	point matching objective, equation (1)
$\{m_{jk}\}$	match matrix variables
$\{\hat{m}_{jk}\}$	match matrix variables including the slacks (see Figure 1)
$\{Q_{jk}\}$	partial derivative of E_{2D} with respect to m_{jk}
I_0	maximum # of iterations allowed at each value of the control parameter, β
I_1	maximum # of iterations allowed for Sinkhorn's method
	(back and forth row and column normalizations)

Table 1: Variable and constant definitions for the point matching algorithm

For the experiments conducted in Section 5.2 on 2D point sets the following values for the constants were used: $\beta_0 = .00091$, $\beta_f = .2$, $\beta_r = 1.075$, $\gamma_0 = .44$, $\alpha = .03$, $I_0 = 4$, and $I_1 = 30$. These values were determined by trial and error. The criterion for convergence for step D was:

$$\sum_{j=1}^{J} \sum_{k=1}^{K} |m_{jk}^0 - m_{jk}| < \epsilon_2$$

For step B the convergence criterion was

$$|a^0 - a| + |b^0 - b| + |c^0 - c| + |t^0 - t| + |\Theta^0 - \Theta| < \epsilon_1$$

where $\{a, b, c, \Theta, t\}$ are the affine parameters in (1). In the above the superscript 0 indicates the intial value of the variable at entry into an iterative loop. In step B $\epsilon_1 = .005$ and in step D $\epsilon_2 = .05$ for the experiments in Section 5.2.

The stopping criterion in step D of the algorithm is a test for convergence as well as a check to see if the maximum number of iterations have been exceeded. This is more efficient because in practice it's unnecessary to always have an exactly doubly stochastic matrix—something close to one works well also.

In the above algorithm we also are decreasing the value of γ, our regularization parameter for the scale and shear components of the affine, as the algorithm converges towards a solution. This is because the scale and shear variables need only be bounded in the beginning of the algorithm, when the initial range of possible values is unrestricted. As the algorithm progresses the estimates of pose and correspondence become better and better, narrowing the range of possible values and thereby gradually eliminating the need to bound these variables.

In the following we let $W \stackrel{\text{def}}{=} s(a)Sh_1(b)Sh_2(c)$ using (1). Then our update equations for Θ and t in step E of the algorithm are:

$$\Theta = \tan^{-1} \frac{\sum_{jk} m_{jk}((X_{j2}-t_2)W_{k1}-(X_{j1}-t_1)W_{k2})}{\sum_{jk} m_{jk}((X_{j1}-t_1)W_{k1}-(X_{j2}-t_2)W_{k2})}$$

$$t_1 = \frac{\sum_{jk} m_{jk}(X_{j1}-[AY]_{k1})}{\sum_{jk} m_{jk}}$$

$$t_2 = \frac{\sum_{jk} m_{jk}(X_{j2}-[AY]_{k2})}{\sum_{jk} m_{jk}}$$

To update a, b and c in step E, the first and second partial derivatives of (1) are calculated with respect to a, b, and c and these first and second partial derivatives are then used in Newton's method to calculate the fixed points.

3.4 Constructing an Objective that Enforces the Constraints

The dynamics of the algorithm may also be motivated by taking the objective function (1), described above and adding an $x \log x$ barrier function and Lagrange multipliers to enforce the constraints. The point matching objective (1) becomes:

$$\hat{E}_{2D}(m,t,A) = \sum_{j=1}^{J}\sum_{k=1}^{K} m_{jk}\|X_j - t - AY_k\|^2 + g(A) - \alpha \sum_{j=1}^{J}\sum_{k=1}^{K} m_{jk}$$
$$+\frac{1}{\beta}\sum_{j=1}^{J+1}\sum_{k=1}^{K+1} m_{jk}(\log m_{jk} - 1) + \sum_{j=1}^{J}\mu_j(\sum_{k=1}^{K+1} m_{jk} - 1) + \sum_{k=1}^{K}\nu_k(\sum_{j=1}^{J+1} m_{jk} - 1) \qquad (2)$$

In the above we are looking for a saddle point by minimizing with respect to m, A and t and maximizing with respect to μ and ν, the Lagrange multipliers.

The $x \log x$ term is a barrier function (also called an entropy term in statistical physics), which serves to push the minimum of the objective away from the discrete points. It convexifies the objective, with the parameter β controlling the degree of convexity. The objective (2) can be derived from using techniques from statistical physics (Rangarajan et al., 1996; Elfadel and Yuille, 1993; Yuille and Kosowsky, 1994; Van den Bout and Miller III, 1990; Peterson and Soderberg, 1989).

At first glance, the softassign with its use of iterated row and column normalization may appear to be unrelated to the energy function in (2). However, this is not the case. Iterated row and column normalization can be directly related (Rangarajan et al., 1996) to solving for the Lagrange parameters (μ and ν) in (2). To see this, examine the fixed point solution for m in the above objective:

$$m_{jk} = \exp\left(\beta\left(Q_{jk} - \mu_j - \nu_k\right)\right)$$

where (as before)

$$Q_{jk} \stackrel{\text{def}}{=} -\frac{\partial E_{2D}}{\partial m_{jk}} = -(\|X_j - t - AY_k\|^2 - \alpha)$$

The fixed point equation contains the two (as yet undetermined) Lagrange parameters μ and ν. The structure of the fixed point solution allows a Lagrange parameter updating scheme where all the μ are updated, followed by all the ν. Let the $(n+1)^{\text{th}}$ update of the Lagrange parameter μ be associated with the $(2n+1)^{\text{th}}$ update of m and the n^{th} update of the Lagrange parameter ν be associated with the $2n^{\text{th}}$ update of m. Now,

$$m_{jk}^{(2n+1)} = \exp\left(\beta\left(Q_{jk} - \mu_j^{(n+1)} - \nu_k^{(n)}\right)\right), \text{ and} \tag{3}$$

$$m_{jk}^{(2n)} = \exp\left(\beta\left(Q_{jk} - \mu_j^{(n)} - \nu_k^{(n)}\right)\right). \tag{4}$$

Taking ratios, we get

$$\frac{m_{jk}^{(2n)}}{m_{jk}^{(2n+1)}} = \exp\left(-\beta\left[\mu_j^{(n)} - \mu_j^{(n+1)}\right)\right]). \tag{5}$$

Setting the derivative of the energy function in (2) w.r.t μ to zero ($\frac{\partial \hat{E}_{\text{2D}}}{\partial \mu} = 0$), we solve for the row constraint:

$$\sum_k M_{jk}^{(2n+1)} = 1 \Rightarrow \exp\left(\beta \mu_j^{(n+1)}\right) = \sum_k \exp\left(\beta\left(Q_{jk} - \nu_k^{(n)}\right)\right). \tag{6}$$

From (4), (5), and (6), we get

$$m_{jk}^{(2n+1)} = \frac{m_{jk}^{(2n)}}{\sum_k m_{jk}^{(2n)}}. \tag{7}$$

We have shown that the μ update can be replaced by row normalization of m. A similar relationship is obtained between the ν update and column normalization. Note that Q_{jk} remains constant during the row and column normalizations. We have demonstrated a straightforward connection between our constraint energy function (2) and Sinkhorn's theorem: solving for the Lagrange parameters in (2) is *identical* to iterated row and column normalization in the softassign.

The row normalization, column normalization, and geometric update phases of this algorithm can also be formalized as a "clocked" objective function, related to (2), which optimizes different subsets of the variables in different clock phases (Rangarajan et al., 1996).

In the above objective we have relaxed one of our original constraints, $\forall jk\ m_{jk} \in \{0,1\}$ to $\forall jk\ m_{jk} \in [0,1]$, i.e. our binary valued correspondence matrix has become a real valued or fuzzy correspondence matrix which now permits partial matches, as long as the sum of these partial matches (across any row or column) adds to one. The degree of fuzziness of the correspondence matrix is controlled by the β parameter. In all the experiments in this paper we have permitted the final correspondence matrix to be fuzzy—however, if an application needed a solution in the form of a binary valued correspondence matrix a simple post processing heuristic could be added to convert the fuzzy real valued matrix into a discrete matrix.

4 3D with Rotation and Translation

The second algorithm solves the 3D-3D pose estimation problem with unknown correspondence. Given two sets of 3D points $\{X_j\}$ and $\{Y_k\}$ find the rotation R, translation T, and correspondence m that minimize

$$E_{3D}(m,T,R) = \sum_{j=1}^{J}\sum_{k=1}^{K} m_{jk}\|X_j - T - RY_k\|^2 - \alpha\sum_{j=1}^{J}\sum_{k=1}^{K} m_{jk}$$

with the same constraint on the correspondence matrix m as in 2D affine matching. Note that there is no regularization term for the $\{T,R\}$ parameters. The major difference from the 2-D case is the absence of the shear and scale parameters.

This structure of this algorithm very similar to the one described for 2D point. The step for updating m—the softassign (step C in the pseudocode) is the same as in 2D affine matching. In the routine corresponding to step E in the 2D case, m is fixed, and we have a labeled 3D to 3D pose estimation problem which is formulated as a weighted least squares problem and solved using a dual number quaternion representation for rotation and translation according to the method outlined in (Walker et al., 1991) and summarized in the next paragraph. The pseudocode for the 3D point matching algorithm is as follows:

Initialize T, R to zero, β to β_0, $\hat{m}_{jk}$ to $(1+\epsilon)$

Begin A: Do A until $(\beta \geq \beta_f)$

Begin B: Do B until m converges or # of iterations $> I_0$

Begin C (update correspondence parameters by softassign):

$Q_{jk} \leftarrow -\frac{\partial E_{2D}}{\partial m_{jk}}$

$m^0_{jk} \leftarrow \exp(\beta Q_{jk})$

Begin D: Do D until $\hat{m}$ converges or # of iterations $> I_1$

Update $\hat{m}$ by normalizing across all rows:

$\hat{m}^1_{jk} \leftarrow \frac{\hat{m}^0_{jk}}{\sum_{k=1}^{K+1} \hat{m}^0_{jk}}$

Update $\hat{m}$ by normalizing across all columns:

$\hat{m}^0_{jk} \leftarrow \frac{\hat{m}^1_{jk}}{\sum_{j=1}^{J+1} \hat{m}^1_{jk}}$

End D

End C

Begin E (update pose parameters using Walker et al.'s method):

Update R, T as described below.

End E

End B

$\beta \leftarrow \beta_r \beta$

End A

In Walker et al.'s method for solving the absolute orientation problem the rotation and translation are represented by a dual number quaternion (r, s),$r^t r = 1$,$r^t s = 0$ which corresponds to a screw coordinate transform. The rotation can be written as $R(r) = W(r)^t P(r)$ and the translation as $W(r)^t s$, where

$$\begin{aligned} r &= (r_1, r_2, r_3, r_4)^t \\ W(r) &= \begin{pmatrix} r_4 & r_3 & -r_2 & r_1 \\ -r_3 & r_4 & r_1 & r_2 \\ r_2 & -r_1 & r_4 & r_3 \\ -r_1 & -r_2 & -r_3 & r_4 \end{pmatrix} \end{aligned}$$

$$P(r) = \begin{pmatrix} r_4 & r_3 & -r_2 & r_1 \\ -r_3 & r_4 & r_1 & r_2 \\ r_2 & -r_1 & r_4 & r_3 \\ -r_1 & -r_2 & -r_3 & r_4 \end{pmatrix}$$

$$P(r) = \begin{pmatrix} r_4 & -r_3 & r_2 & r_1 \\ r_3 & r_4 & -r_1 & r_2 \\ -r_2 & r_1 & r_4 & r_3 \\ -r_1 & -r_2 & -r_3 & r_4 \end{pmatrix}$$

Using these representations, the objective function becomes

$$E_{3D} = \sum_{j=1}^{J} \sum_{k=1}^{K} m_{jk} \| x_j - W(r)^t s - W(r)^t P(r) y_k \|^2$$

where $x_j = (X_j, 0)^t$ and $y_k = (Y_k, 0)^t$ are the quaternion representations of X_j and Y_k, respectively. Using the properties that $P(a)b = W(b)a$ and $P(a)^t P(a) = W(a)^t W(a) = (a^t a)I$, the objective function can be formulated as minimizing

$$E_{3D} = r^t C_1 r + s^t C_2 s + s^t C_3 r \tag{8}$$

subject to $r^t r = 1$ and $r^t s = 0$ where

$$\begin{aligned} C_1 &= -\sum_{j=1}^{J} \sum_{k=1}^{K} m_{jk} P(y_k)^t W(x_j) \\ C_2 &= \frac{1}{2} \sum_{j=1}^{J} \sum_{k=1}^{K} m_{jk} I \\ C_3 &= \sum_{j=1}^{J} \sum_{k=1}^{K} m_{jk} (W(x_j) - P(y_k)) \end{aligned}$$

With this new representation, all the constraint information, including the current fuzzy estimate of the correspondence m is absorbed into the three 4-by-4 matrices C_1, C_2, C_3 in (8). The objective function is minimized by (r^*, s^*) where r^* is the largest eigenvalue of $C_3^t C_2^{-1} C_3 - \frac{1}{2}(C_1 + C_1^t)$ and $s^* = -2C_2^{-1} C_3 r$. (r^*, s^*) is then used to determine the rotation and translation.

5 Experimental Results

In this section we provide results for the 2D matching problems, including over thirty thousand experiments on randomly generated point sets as well as experiments on point sets generated from handwritten characters.

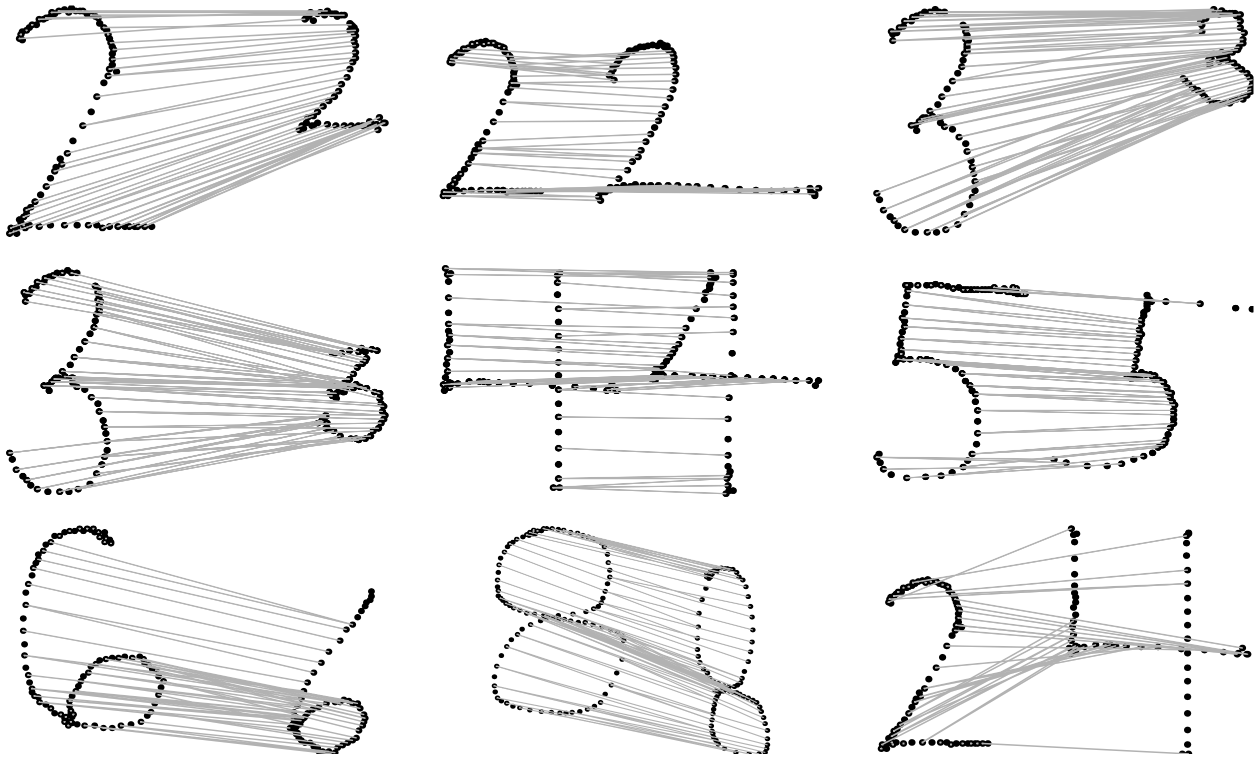


Figure 4: Correspondence of digits

5.1 Handwritten Character Data

Our first experiment used point sets generated from handwritten characters. The handwritten characters were created using an X-windows tool which enables us to draw an image on the screen with the mouse and writing pad. The contours of the images were discretized and are expressed as a set of points in the plane. In the experiments below, we generated 70 points per character on average.

The inputs to the point matching algorithm were the x-y coordinates generated by the drawing program. No other pre-processing was done. The output was a correspondence matrix and a pose. In Figures 4 and 5, the correspondences found between several images drawn in this fashion are shown.To make the actual point matches easier to see, we have drawn the correspondences only for every other image point.

In Figure 4 are several examples, using different digits, of matches between two handwritten versions of the same digit. Despite large variations between two versions of the same digit, the algorithm found accurate correspondences. Excellent correspondences are found despite large differences in scale. The correspondence was good between distorted digits, as in 3 and 6, or between

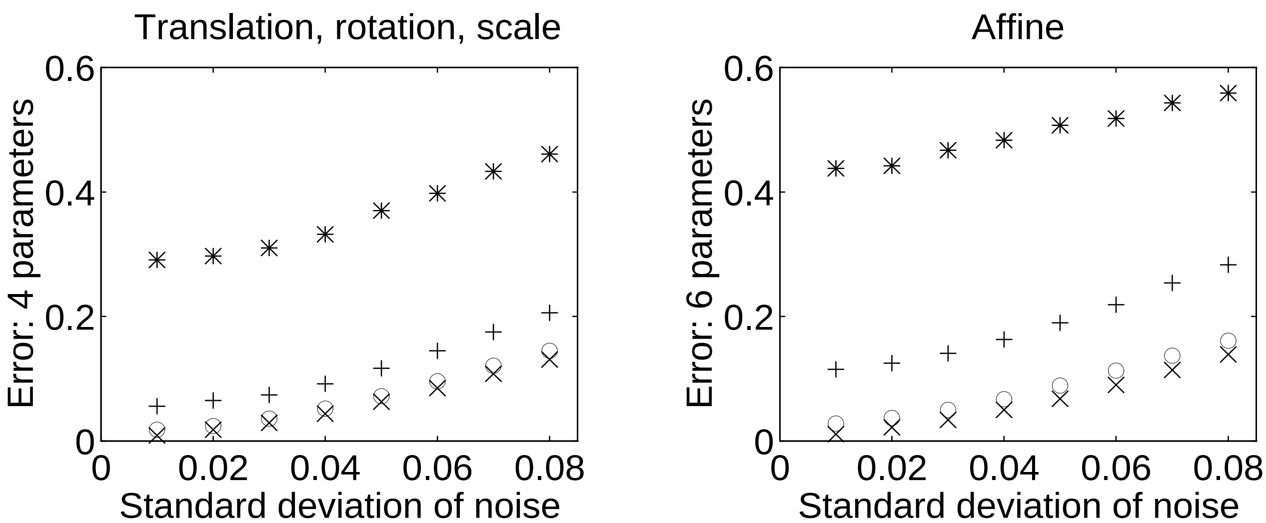
Figure 5: Correspondence: "a" found in "cat", "o" found in "song"

different forms of a digit as in 4 and 3. We also provided an example of trying to finding the correspondence between two different digits, a 2 and a 4, which as to be expected, did not work well.

In another experiment (Figure 5), individual letters were correctly identified within words. Here, no pre-processing to segment the cursive word into letters was done. The correspondence returned by the point matching algorithm by itself was good enough for identification. Even similar letters were differentiated, for example the "a" in cat was correctly identified even though the "c" has a similar shape and the "o" was correctly identified in "song," despite the similarity of the "s." The time to recognize each character (which could contain over 100 points) was on the order of a minute on a Silicon Graphics workstation with a R4400 processor.

5.2 Randomly generated point sets

In the second set of experiments (Figure 6), randomly generated point sets were matched. In each trial one point set, the model, was created by randomly generating with a uniform distribution, 50 points on a grid of unit area. This point set, the model, was then used to create another point set, the image, by adding noise to the model points, deleting and adding model points and finally remapping the model points in 2D space by applying a randomly generated affine transform. The image point set created in this manner was then matched to the model point set, using our algorithm. The pose parameters, $\{a, b, c, \Theta, t\}$ of equation (1), returned by the algorithm were then compared to the original pose parameters used in the affine transform to create the image point set by remapping the original model point set. The difference between the two sets of pose parameters was reported as the error returned by the algorithm in Figure 6. All parameters were selected in a random fashion and over thirty thousand different trials were conducted.



x : $p_d = 0.0, p_s = 0.0,$ o : $p_d = 0.1, p_s = 0.1$

$+$: $p_d = 0.3, p_s = 0.1,$ $*$: $p_d = 0.5, p_s = 0.1$

Figure 6: 2D results for synthetic data. X axis is σ, the noise of independent Gaussian jitter.

To create an image point set from a model point set, independent Gaussian noise $N(0, \sigma)$ is added to each of the model points creating jittered image points. Then a fraction, p_d, of the points are deleted, and a fraction, p_s, of spurious points are added, randomly (with a uniform distribution) on the unit square. Finally a randomly generated (with a uniform distribution) affine transformation, is applied to the set of points resulting in the new image point set.

Two sets of experiments, using different transformations were conducted. In the first set (right plot in Figure 6) a full affine transformation was applied, in the second set (left plot in Figure 6) a transform composed only of translation, rotation and scale was used. The algorithm detailed in this paper can be trivially modified to find only scale, rotation and translation (initialize the shear variables, $\{b, c\}$ to zero and never update them in step E of the algorithm). Therefore the transformations we considered in the two sets of experiments were $\hat{A}, t \rightarrow$ (Translation, rotation, scale) and the full affine transformation, $A \rightarrow, t$ (Translation, rotation, scale, and two components of shear). The transformation parameters, $\{t_x, t_y, \theta, a, b, c\}$ were from selected from the following ranges: $-0.5 < t_x, t_y < 0.5$, $-27^\circ < \theta < 27^\circ$, $0.5 \leq e^a \leq 2$ where a is the scale parameter, and $0.7 \leq e^b, e^c \leq 1/0.7$ where b, c are the parameters for the two shears (except in the second set of experiments where b, c were set to zero). Each of the parameters was chosen independently and uniformly from the possible ranges.

The error measure (the difference between the pose parameters returned by the algorithm and the pose parameters used to generate the image point set from the model point set) was computed with $e_z = 3|\frac{z^{actual} - z^{estimate}}{width_z}|$, where e_z is the error measure for parameter z and $width_z$ is the range of possible values for z. z is one of the pose parameters, $\{a, b, c, \Theta, t\}$, z^{actual} is the value of the pose parameter used to generated the image point set from the model point set, $z^{estimate}$ is the value of the pose parameter returned by the algorithm and $width_z$ is computed from the ranges reported in the previous paragraph. Dividing by $width_z$ is preferable to dividing by z^{actual}, which incorrectly weights small z^{actual} values. Multiplying by 3 normalizes the error measure to give an expected value of one in the case where z^{actual} and $z^{estimate}$ are simply selected at random from a uniform distribution of width $width_z$. The reported error (y axes of Figure 6) is the average error over all the parameters (4 or 6).

The time to recover the correspondence and pose for a problem instance of 50 points is about 50 seconds on a Silicon Graphics workstation with a R4400 processor. By varying parameters such as the annealing rate (β_r) or stopping criterion, this can be reduced to about 20 seconds with some degradation in accuracy. For each trial combinations of $\sigma \in \{0.01, 0.02, \ldots, 0.08\}$ and $p_d \in \{0\%, 10\%, 30\%, 50\%\}$ and $p_s \in \{0\%, 10\%\}$ were used.

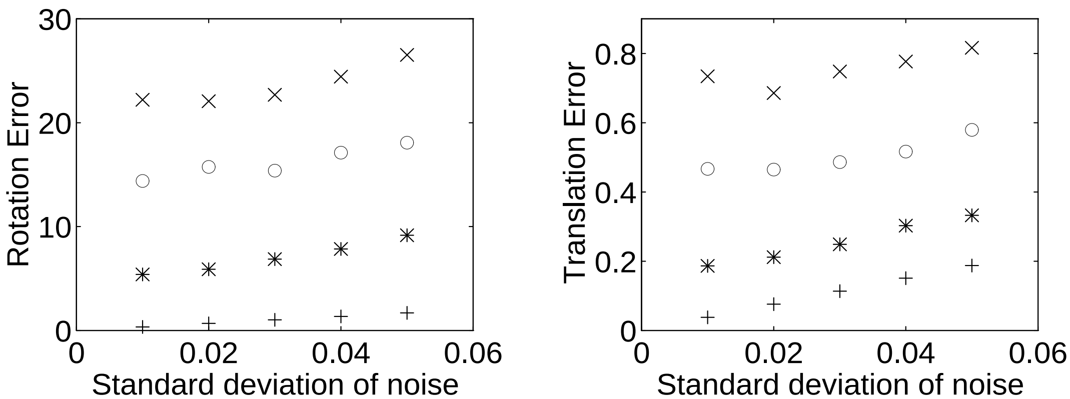
Results are reported separately for transformations $\hat{A}, t$ and A, t. For each combination of (σ-p_d-p_s) 500 test instances were generated. Each data point in Figures 6.a and 6.b represents the average error measure for 500 trial runs, one trial for each test instance; all data points represent a total of 32,000 test instances. Raising the noise in the form of independent Gaussian jitter and/or missing and extra points increases the error measure monotonically. As expected, the transformation $\hat{A}$ has better results than the affine transformation A. The parameter values used in these experiments are reported in Section 3.3.

5.3 Randomly generated point sets: 3D

The experimental setup for 3D point matching is similar to the setup for 2D (random) point matching described in Section 5.2. Each experimental trial for 3D point matching involves generating a random 3D point set as the model point set, and then generating an image point set by applying a random transformation, adding noise of independent Gaussian jitter and then randomly adding and deleting points.

20 points are generated from a uniform distribution within a unit cube. The parameters for the transformation are generated as follows: The three Euler angles for determining R are selected from

a uniform distribution $U[20°, 70°]$. Translation parameters T_x, T_y, T_z are selected from a uniform distribution $U[2.5, 7.5]$. Gaussian noise $N(0, \sigma)$ is added to the points. The objective then is to recover the three translation and three rotation parameters and to find the correspondence between this and the original point set. The rotation errors in degrees and the translation errors in units are summarized in Figure 7. These plots show the average differences between the rotations and translations returned by the algorithm and the rotations and translations used to generate the image point sets. For each combination of (σ-p_d-p_s) 150 trials were run. These experiments were conducted with the following parameter values: $\beta_0 = .01/S$, $\beta_r = 1.053$, $\beta_f = 100/S$, $I_0 = 10$ and $I_1 = 30$ where $S = \frac{1}{JK}\sum_{jk}\|X_j - Y_k\|^2$, the mean distance between points. These experiments on sets of 20 points took under 5 seconds on a Silicon Graphics workstation.



x : $p_d = 0.0, p_s = 0.0$, o : $p_d = 0.1, p_s = 0.1$

$+$: $p_d = 0.2, p_s = 0.2$, $*$: $p_d = 0.3, p_s = 0.3$

Figure 7: 3D results for synthetic data. Rotation is in degrees; the translation error should be compared to translations $\in [2.5, 7.5]$. X axis is σ, the noise of independent Gaussian jitter.

6 Conclusion

We have developed a new, fast, robust and easily implementable algorithm for solving 2D and 3D pose estimation and correspondence problems. It is *new*—the algorithm incorporates a novel optimization technique, the *softassign* which has recently emerged from the neural network/statistical physics framework. The softassign, which allows optimization problems requiring two-way (assignment) constraints to be solved without the use of any penalty terms in the objective functions, is applied for the first time to point matching. It is *fast*—we are able to match sets of 50 or more points (including missing and spurious points) in a few seconds. Many other point matching methods, when trying to find affine transformations between noisy point sets with missing and extra points are much less efficient: For example see tree-pruning (Baird, 1984), alignment (Ullman, 1989), neural network (Gee et al., 1993), Hough transforms (Grimson, 1990), geometric hashing (Lamdan et al., 1988), and pose clustering (Stockman, 1987). It is *robust*—its use of slack variables permit the handling of outliers in a statistically robust manner. Moreover, a large number of experiments demonstrate it can tolerate a high degree of noise and still recover accurate pose parameters. It is *easy to implement*. The optimization techniques are simple; the correspondence parameters are calculated using the softassign and the pose parameters are calculated using coordinate descent. No gradient descent methods are used, eliminating the need for line searches and/or gradient projections. No specialized pre-packaged optimization software or routines are required.

Many promising directions for enhancement of this algorithm exist. The affine transform need not be decomposed; it can be solved for directly in the algorithm, simplifying the procedure even more (Pappu et al., 1996). Instead of just using location information, feature vectors may be attached to each point, indicating such low-level visual attributes as edges, curves, color, etc. (Gold, 1995; Pappu et al., 1996). Multiple affine transformations may be applied to the same image, allowing a decomposition of the object into parts (Gold, 1995; Pappu et al., 1996; Mjolsness, 1991; Mjolsness, 1994). Finally the algorithm may be used as a distance measure and incorporated within a clustering algorithm to learn prototypical object shapes (Gold et al., 1996; Gold, 1995).

Acknowledgements

This work was supported by AFOSR Grant F49620-92-J-0465, ONR/DARPA grant N00014-92-J-4048 and the Yale Neuroengineering and Neuroscience Center.

References

Baird, H. (1984). *Model-Based Image Matching Using Location*. MIT Press, Cambridge, MA.

Ballard, D. H. (1981). Generalized hough transform to detect arbitrary patterns. *IEEE Trans. Patt. Anal. Mach. Intell.*, 13(2):111–122.

Bertsekas, D. P. and Tsitsiklis, J. N. (1989). *Parallel and Distributed Computation: Numerical Methods*. Prentice-Hall, Englewood Cliffs, NJ.

Besl, P. J. and McKay, N. D. (1992). A method for registration of 3-D shapes. *IEEE Trans. Patt. Anal. Mach. Intell.*, 14(2):239–256.

Black, M. and Rangarajan, A. (1996). On the unification of line processes, outlier rejection, and robust statistics with applications to early vision. *International Journal of Computer Vision*, 19(1):57–91.

Bridle, J. S. (1990). Training stochastic model recognition algorithms as networks can lead to maximum mutual information estimation of parameters. In Touretzky, D. S., editor, *Advances in Neural Information Processing Systems 2*, pages 211–217, San Mateo, CA. Morgan Kaufmann.

Brunelli, R. and Poggio, T. (1993). Face recognition: Features versus templates. *IEEE Trans. Patt. Anal. Mach. Intell.*, 15(10):1042–1052.

Califano, A. and Mohan, R. (1994). Multidimensional indexing for recognizing visual shapes. *IEEE Trans. Patt. Anal. Mach. Intell.*, 16(4):373–392.

Chvatal, V. (1983). *Linear Programming*. W. H. Freeman and Company, New York.

Duda, R. and Hart, P. (1973). *Pattern Classification and Scene Analysis*. Wiley, New York, NY.

Elfadel, I. M. and Yuille, A. L. (1993). Mean-field phase transitions and correlation functions for Gibbs random fields. *J. Math. Imaging Vision*, 3(2):167–186.

Gee, A. H., Aiyer, S., and Prager, R. W. (1993). An analytical framework for optimizing neural networks. *Neural Networks*, 6:79–97.

Geiger, D. and Girosi, F. (1991). Parallel and deterministic algorithms from MRFs: Surface reconstruction. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 13(5):401–412.

Geiger, D. and Yuille, A. L. (1991). A common framework for image segmentation. *Intl. Journal of Computer Vision*, 6(3):227–243.

Girosi, F., Jones, M., and Poggio, T. (1995). Regularization theory and neural network architectures. *Neural Computation*, 7:219–269.

Gold, S. (1995). *Matching and Learning Structural and Spatial Representations with Neural Networks.* PhD thesis, Yale University.

Gold, S. and Rangarajan, A. (1996a). A graduated assignment algorithm for graph matching. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(4):377–388.

Gold, S. and Rangarajan, A. (1996b). Softassign versus softmax: Benchmarks in combinatorial optimization. In Touretzky, D. S., Mozer, M. C., and Hasselmo, M. E., editors, *Advances in Neural Information Processing Systems 8*, pages 626–632. MIT Press, Cambridge, MA.

Gold, S., Rangarajan, A., and Mjolsness, E. (1996). Learning with preknowledge: clustering with point and graph matching distance measures. *Neural Computation*, 8(4):787–804.

Grimson, E. (1990). *Object Recognition by Computer: The Role of Geometric Constraints.* MIT Press, Cambridge, MA.

Grimson, E. and Lozano-Perez, T. (1987). Localizing overlapping parts by searching the interpretation tree. *IEEE Trans. Patt. Anal. Mach. Intell.*, 9:468–482.

Hinton, G., Williams, C., and Revow, M. (1992). Adaptive elastic models for hand-printed character recognition. In Moody, J., Hanson, S., and Lippmann, R., editors, *Advances in Neural Information Processing Systems 4*. Morgan Kaufmann, San Mateo, CA.

Hopfield, J. J. and Tank, D. (1985). 'Neural' computation of decisions in optimization problems. *Biological Cybernetics*, 52:141–152.

Hummel, R. and Wolfson, H. (1988). Affine invariant matching. In *Proceedings of the DARPA Image Understanding Workshop*, pages 351–364, Cambridge, MA.

Huttenlocher, D. P., Klanderman, G. A., and Rucklidge, W. J. (1993). Comparing images using the Hausdorff distance. *IEEE Trans. Patt. Anal. Mach. Intell.*, 15(9):850–863.

Jacobs, D. W. (1994). Generalizing invariants for 3-D to 2-D matching. In Mundy, J. L., Zisserman, A., and Forsyth, D., editors, *Springer Lecture Notes in Computer Science*, volume 825, pages 415–434.

Kamgar-Parsi, B. and Kamgar-Parsi, B. (1990). On problem solving with Hopfield networks. *Biological Cybernetics*, 62:415–423.

Kass, M., Witkin, A., and Terzopoulos, D. (1988). Snakes: Active contour models. *International Journal of Computer Vision*, 1(4):321–331.

Kosowsky, J. J. and Yuille, A. L. (1994). The invisible hand algorithm: Solving the assignment problem with statistical physics. *Neural Networks*, 7(3):477–490.

Lamdan, Y., Schwartz, J., and Wolfson, H. (1988). Object recognition by affine invariant matching. *IEEE Conf. Comp. Vision, Patt. Recog.*, pages 335–344.

Li, S. Z. (1992). Matching: invariant to translations, rotations and scale changes. *Pattern Recognition*, 25:583–594.

Lu, C. P. and Mjolsness, E. (1994). Two-dimensional object localization by coarse-to-fine correlation matching. In Cowan, J., Tesauro, G., and Alspector, J., editors, *Advances in Neural Information Processing Systems 6*, pages 985–992. Morgan Kaufmann, San Francisco, CA.

Mjolsness, E. (1991). Bayesian inference on visual grammars by neural nets that optimize. Technical Report YALEU/DCS/TR-854, Department of Computer Science, Yale University.

Mjolsness, E. (1994). Connectionist grammars for high-level vision. In Honavar, V. and Uhr, L., editors, *Artificial Intelligence and Neural Networks: Steps Toward Principled Integration*, pages 423–451. Academic Press, Inc., San Diego, CA.

Mjolsness, E. and Garrett, C. (1990). Algebraic transformations of objective functions. *Neural Networks*, 3:651–669.

Olson, C. F. (1995). Probabilistic indexing for object recognition. *IEEE Trans. Patt. Anal. Mach. Intell.*, 17(5):518–521.

Papadimitriou, C. and Steiglitz, K. (1982). *Combinatorial Optimization.* Prentice-Hall, Inc., Englewood Cliffs, NJ.

Pappu, S., Gold, S., and Rangarajan, A. (1996). A framework for nonrigid matching and correspondence. In Touretzky, D. S., Mozer, M. C., and Hasselmo, M. E., editors, *Advances in Neural Information Processing Systems 8*, pages 795–801. MIT Press, Cambridge, MA.

Peterson, C. and Soderberg, B. (1989). A new method for mapping optimization problems onto neural networks. *Intl. Journal of Neural Systems*, 1(1):3–22.

Ranade, S. and Rosenfeld, A. (1980). Point pattern matching by relaxation. *Pattern Recognition*, 12:269–275.

Rangarajan, A., Gold, S., and Mjolsness, E. (1996). A novel optimizing network architecture with applications. *Neural Computation*, 8(5):1041–1060.

Sclaroff, S. and Pentland, A. P. (1995). Modal matching for correspondence and recognition. *IEEE Trans. Patt. Anal. Mach. Intell.*, 17(6):545–561.

Scott, G. and Longuet-Higgins, C. (1991). An algorithm for associating the features of two images. *Proc. Royal Society of London*, B244:21–26.

Shapiro, L. and Brady, J. (1992). Feature-based correspondence: an eigenvector approach. *Image and Vision Computing*, 10:283–288.

Simic, P. D. (1991). Constrained nets for graph matching and other quadratic assignment problems. *Neural Computation*, 3:268–281.

Sinkhorn, R. (1964). A relationship between arbitrary positive matrices and doubly stochastic matrices. *Ann. Math. Statist.*, 35:876–879.

Stockman, G. (1987). Object recognition and localization via pose clustering. *Computer Vision, Graphics, and Image Processing*, 40.

Ton, J. and Jain, A. (1989). Registering Landsat images by point matching. *IEEE Trans. Geo. Rem. Sens.*, 27(5):642–651.

Ullman, S. (1989). Aligning pictorial descriptions: An approach to object recognition. *Cognition*, 32(3):193–254.

Umeyama, S. (1993). Parameterized point pattern matching and its application to recognition of object families. *IEEE Trans. Patt. Anal. Mach. Intell.*, 15(1):136–144.

Van den Bout, D. E. and Miller III, T. K. (1990). Graph partitioning using annealed networks. *IEEE Trans. Neural Networks*, 1(2):192–203.

Vinod, V. and Ghose, S. (1993). Point matching using asymmetric neural networks. *Pattern Recognition*, 26:1207–1214.

Walker, M. W., Shao, L., and Volz, R. (1991). Estimating 3-D location parameters using dual number quaternions. *CVGIP: Image Understanding*, 54(3):358–367.

Weinshall, D. and Tomasi, C. (1995). Linear and incremental acquisition of invariant shape models from image sequences. *IEEE Trans. Patt. Anal. Mach. Intell.*, 17(5):512–517.

Wilson, G. V. and Pawley, G. S. (1988). On the stability of the traveling salesman problem algorithm of Hopfield and Tank. *Biological Cybernetics*, 58:63–70.

Yuille, A. and Hallinan, P. (1992). Deformable templates. In Blake, A. and Yuille, A., editors, *Active Vision*. MIT Press, Cambridge, MA.

Yuille, A. L. (1990). Generalized deformable models, statistical physics, and matching problems. *Neural Computation*, 2(1):1–24.

Yuille, A. L. and Kosowsky, J. J. (1994). Statistical physics algorithms that converge. *Neural Computation*, 6(3):341–356.