SoftPOSIT: Simultaneous Pose and Correspondence Determination

Philip David[1,2], Daniel DeMenthon[3], Ramani Duraiswami[4], and Hanan Samet[1]
{phild@cs,daniel@cfar,ramani@umiacs,hjs@cs}.umd.edu *

[1] Department of Computer Science, University of Maryland, College Park, MD 20742
[2] Army Research Laboratory, 2800 Powder Mill Road, Adelphi, MD 20783-1197
[3] Language and Media Processing Laboratory, Institute for Advanced Computer Studies, University of Maryland, College Park, MD 20742
[4] Perceptual Interfaces and Reality Laboratory, Institute for Advanced Computer Studies, University of Maryland, College Park, MD 20742

Abstract. The problem of pose estimation arises in many areas of computer vision, including object recognition, object tracking, site inspection and updating, and autonomous navigation using scene models. We present a new algorithm, called *SoftPOSIT*, for determining the pose of a 3D object from a single 2D image in the case that correspondences between model points and image points are unknown. The algorithm combines Gold's iterative SoftAssign algorithm [19, 20] for computing correspondences and DeMenthon's iterative POSIT algorithm [13] for computing object pose under a full-perspective camera model. Our algorithm, unlike most previous algorithms for this problem, *does not* have to hypothesize small sets of matches and then verify the remaining image points. Instead, *all* possible matches are treated identically throughout the search for an optimal pose. The performance of the algorithm is extensively evaluated in Monte Carlo simulations on synthetic data under a variety of levels of clutter, occlusion, and image noise. These tests show that the algorithm performs well in a variety of difficult scenarios, and empirical evidence suggests that the algorithm has a run-time complexity that is better than previous methods by a factor equal to the number of image points. The algorithm is being applied to the practical problem of autonomous vehicle navigation in a city through registration of a 3D architectural models of buildings to images obtained from an on-board camera.

Keywords: Object recognition, autonomous navigation, POSIT, SoftAssign

1 Introduction

We present an algorithm for solving the *model-to-image registration problem*, which determines the position and orientation (the *pose*) of a 3D object with respect to a camera coordinate system given a model of the object with 3D reference points and a single 2D image of these points. We assume no additional information to constrain the pose of the object or the correspondences. This is also known as the *simultaneous pose and correspondence problem.*

Automatic registration of 3D models to images is important for many applications, including object recognition and tracking, site inspection and updating, and autonomous

* Support of NSF grants EAR-9905844, IIS-0086162, and IIS-9987944 is gratefully acknowledged.

navigation using scene models. The problem is difficult because it requires solution of two coupled problems, *correspondence* and *pose*, each easy to solve only if the other has been solved first:

1. Solving the *pose* problem consists of finding the rotation and translation of the object with respect to the camera coordinate system. Given matching model and image features, one can determine the pose that best aligns those matches. For 3 to 5 matches, the pose can be found in closed-form by solving polynomial equations [17, 24, 37]. For six or more matches, linear and nonlinear approximate methods are generally used [13, 16, 23, 25, 29].
2. Solving the *correspondence* problem requires matching image and model features. If the object pose is known, one can determine such matches. Projecting the model with known pose into the original image, one can match features that project sufficiently close to an image feature. This is the approach typically taken for pose verification [22].

The classic approach to solving these coupled problems is the hypothesize-and-test approach [21] where a small set of correspondences are first hypothesized, and the corresponding pose of the object is computed. Using this pose, the model points are back-projected into the image. If the original and back-projected images are sufficiently similar, the pose is accepted; else a new hypothesis is formed, and the process repeated. The best known example of this approach is RANSAC [17] for the case that no information is available to constrain the correspondences. When there are J image points and K model points[1], and three correspondences are used to determine pose, a high probability of success can be achieved by the RANSAC algorithm in $O(KJ^4)$ operations [11].

The problem we address occurs when taking a model-based approach to object-recognition. (The other main approach to object recognition is appearance-based [31], where multiple object views are compared to the image. However, since 3D models are not used, accurate object pose is not recovered.) Many investigators (e.g., [8, 9, 15, 26, 28, 33]) approximate the nonlinear perspective projection via linear affine approximations. This is accurate when the relative depth of object features is small compared to the distance of the object from the camera. Among the pioneer contributions were Baird's tree-pruning method [1], with exponential time complexity for unequal point sets, and Ullman's alignment method [35] with time complexity $O(J^4K^3 \log K)$. Geometric hashing is employed in [28] to determine an object's identity and pose using a hashing metric computed from image features; because the metric must be invariant to camera viewpoint, the method can only be applied for affine camera models [5]. In [12] an approach using a binary search by bisection of pose boxes in two 4D spaces was used, extending the work of [1, 6, 7] on affine transforms, however the method was computationally intensive. The approach of [27] is similar: An initial volume of pose space is guessed, and all correspondences compatible with this volume are accounted for. Then the pose volume is recursively reduced until it can be viewed as a single pose.

Few researchers have addressed the full perspective problem. The object recognition approach of [2] extends the geometric hashing approach by using non-invariant image

[1] Many authors use N and M instead of J and K to denote the numbers of image and model points.

features: off-line training is performed to learn 2D feature groupings associated with a large number of views. An on-line recognition stage then uses new feature groupings to index into a database of learned model-to-image correspondence hypotheses, and these hypotheses are used for pose estimation and verification. In [36], the abstract problem is formalized in a way similar to the present approach, as the optimization of an objective function combining correspondence and pose. However, the correspondence constraints are not represented analytically. Instead, each model feature is explicitly matched to the closest line of sight of the image features. The closest 3D points on the lines of sight are found for each model feature, and then the pose that brings the model features closest to these 3D points is selected; this allows an easier 3D to 3D pose problem to be solved. The process is repeated until a minimum is reached. A randomized pose clustering algorithm is presented in [32] whose time complexity is $O(KJ^3)$. In this approach, instead of testing each hypothesis as it is generated (as in the RANSAC approach), all hypotheses are clustered in a pose space before back-projection and testing. This step is performed only on high probability poses that are determined from the larger clusters.

A method related to ours is presented in [3] that uses random-start local search with a hybrid pose estimation algorithm employing both full and weak perspective models. A steepest descent search in the space of model-to-image line segment correspondences is performed. A weak-perspective algorithm is used in ranking neighboring points in this search space, and a full-perspective algorithm is used to update the model's pose for new correspondence sets. The empirical time complexity is $O(K^2J^2)$.

Our approach, termed the *SoftPOSIT* algorithm, integrates the iterative pose technique called POSIT (Pose from Orthography and Scaling with ITerations) due to DeMenthon and Davis [13], and the iterative 2D to 2D or 3D to 3D correspondence assignment technique called *SoftAssign* due to Gold and Rangarajan [19, 20]. A global objective function is defined that captures the nature of the problem in terms of both pose and correspondence, which are determined *simultaneously* by applying a deterministic annealing schedule and by minimizing this global objective function at each step.

In the following sections, we describe each step of the method, and provide pseudocode for the algorithm. We then evaluate the algorithm using Monte Carlo simulations with various levels of clutter, occlusion and image noise, and finally apply the algorithm to some imagery of a city scene.

2 POSIT Algorithm

We summarize the original POSIT algorithm [13] and then present a variant that performs closed-form minimization of an objective function. This function is modified below to include the simultaneous pose and correspondence problem in one objective.

Consider a pinhole camera of focal length f and an image feature point p with Euclidean and homogeneous coordinates x, y and (wx, wy, w), respectively; p is the perspective projection of 3D point P with homogeneous coordinates $(X, Y, Z, 1)$ in the frame of an object with origin P_0. There is an unknown transformation between the object and the camera coordinates, represented by a rotation matrix $R = [\mathbf{R}_1\ \mathbf{R}_2\ \mathbf{R}_3]^T$ and a translation vector $\mathbf{T} = (T_x, T_y, T_z)$. $\mathbf{R}_1^T$, $\mathbf{R}_2^T$, $\mathbf{R}_3^T$ are the row vectors of R. They are the unit vectors of the camera coordinate system expressed in the model system. $\mathbf{T}$ is the vector from the center of projection O of the camera to the origin P_0 expressed

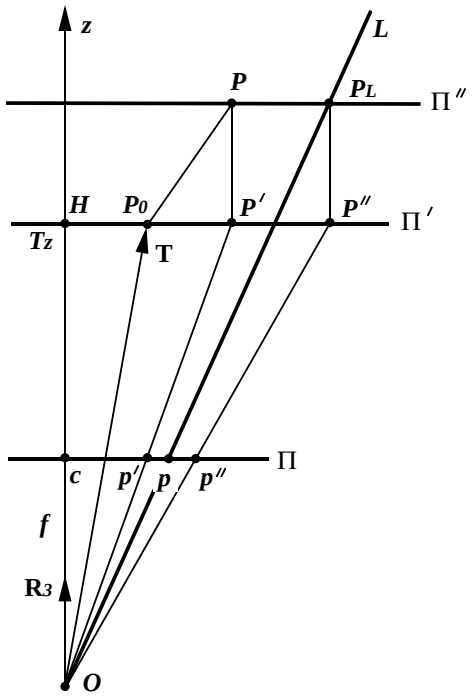


Fig. 1. Geometric interpretation of the POSIT computation. Image point p', the scaled orthographic projection of world point P, is computed by one side of the POSIT equations. Image point p'', the scaled orthographic projection of point P_L on the line of sight of p, is computed by the other side of the equation. The equations are satisfied when the two points are superposed, which requires that the world point P be on the line of sight of image point p. The plane of the figure is chosen to contain the plane of the optical axis and the line of sight L. The points P_0, P, P', and p' are generally out of the plane.

in the camera coordinate system. The coordinates of the projection p are related to the world point P by

$$\begin{bmatrix} wx \\ wy \\ w \end{bmatrix} = \begin{bmatrix} f\mathbf{R}_1^T & fT_x \\ f\mathbf{R}_2^T & fT_y \\ \mathbf{R}_3^T & T_z \end{bmatrix} \begin{bmatrix} \boldsymbol{P_0P} \\ 1 \end{bmatrix},$$

where $\boldsymbol{P_0P} = (X, Y, Z)^T$ is the vector from P_0 to P. The homogeneous image coordinates are defined up to a multiplicative constant; therefore the validity of the equality is not affected if we multiply all elements of the projection matrix by $1/T_z$. Introducing the scaling factor $s = f/T_z$, we obtain

$$\begin{bmatrix} wx \\ wy \end{bmatrix} = \begin{bmatrix} s\mathbf{R}_1^T & sT_x \\ s\mathbf{R}_2^T & sT_y \end{bmatrix} \begin{bmatrix} \boldsymbol{P_0P} \\ 1 \end{bmatrix}, \quad w = \mathbf{R}_3 \cdot \boldsymbol{P_0P}/T_z + 1. \tag{1}$$

In Eq. 1 for w, $\mathbf{R}_3 \cdot \boldsymbol{P_0P}$ represents the projection of $\boldsymbol{P_0P}$ onto the optical axis. When the depth range of the model along the optical axis is small compared to the

model distance, $\mathbf{R}_3 \cdot \boldsymbol{P_0P}$ is small compared to T_z, and $w \simeq 1$. Then, perspective projection is well approximated by *scaled orthographic* projection (SOP) with scaling factor s:

$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} s\mathbf{R}_1^T & sT_x \\ s\mathbf{R}_2^T & sT_y \end{bmatrix} \begin{bmatrix} \boldsymbol{P_0P} \\ 1 \end{bmatrix}. \tag{2}$$

The general perspective equation (1) can be rewritten as

$$\begin{bmatrix} X & Y & Z & 1 \end{bmatrix} \begin{bmatrix} s\mathbf{R}_1 & s\mathbf{R}_2 \\ sT_x & sT_y \end{bmatrix} = \begin{bmatrix} wx & wy \end{bmatrix}. \tag{3}$$

Assume the homogeneous coordinate w for each image point p has been computed at a previous step. We can then calculate wx, wy, and Eq. 3 relates the unknown pose components $s\mathbf{R}_1$, $s\mathbf{R}_2$, sT_x, sT_y, and the known image components wx, wy and known world coordinates X, Y, Z. If we know K world points P_k, $k = 1, \ldots, K$, their image points p_k, and their homogeneous components w_k, we can write two linear systems of size K that can be solved for the unknown components of $s\mathbf{R}_1$, $s\mathbf{R}_2$ and sT_x and sT_y, provided at least four of the points of the model with given image points are noncoplanar. After $s\mathbf{R}_1$ and $s\mathbf{R}_2$ are obtained, we get s, $\mathbf{R}_1$ and $\mathbf{R}_2$, by imposing that $\mathbf{R}_1$ and $\mathbf{R}_2$ be unit vectors, and that $\mathbf{R}_3$ be the cross-product of $\mathbf{R}_1$ and $\mathbf{R}_2$:

$$s = (|s\mathbf{R}_1|\,|s\mathbf{R}_2|)^{1/2}, \quad \mathbf{R}_1 = (s\mathbf{R}_1)/s,\ \mathbf{R}_2 = (s\mathbf{R}_2)/s, \quad \mathbf{R}_3 = \mathbf{R}_1 \times \mathbf{R}_2,$$

$$T_x = (sT_x)/s,\ T_y = (sT_y)/s,\ T_z = f/s.$$

To compute the w_k required in the right-hand side of Eq. (3), we initially set $w_k = 1$ for every point p_k (corresponding to a SOP model). Once we get the pose for this first step, we compute better estimates for the w_k using the expression for w in Eq. (1). Then we can iteratively solve Eqs. (3) again to obtain progressively refined poses. This iteration is stopped when the process becomes stationary.

3 Geometry and Objective Function

We consider a geometric interpretation of POSIT to represent it as minimization of an objective function. Consider, as in Fig. 1, a pinhole camera with center of projection at O, optical axis along Oz, an image plane Π at distance f from O, and an image center at c. Consider an object, the origin of its coordinate system at P_0, an object point P, corresponding image point p, and line of sight L through p. The image point p' is the SOP of object point P. The image point p'' is the SOP of point P_L obtained by shifting P to the line of sight of p in a direction parallel to the image plane.

One can show [14] that the image plane vector from c to p', is

$$\boldsymbol{cp'} = s(\mathbf{R}_1 \cdot \boldsymbol{P_0P} + T_x, \mathbf{R}_2 \cdot \boldsymbol{P_0P} + T_y).$$

In other words, the left-hand side of Eq. (3) represents the vector $\boldsymbol{cp'}$ in the image plane. One can also show [14] that the image plane vector from c to p'' is $\boldsymbol{cp''} = (wx, wy) = w\boldsymbol{cp}$. In other words, the right-hand side of Eq. (3) represents the vector $\boldsymbol{cp''}$ in the image plane. The image point p'' can be interpreted as a correction of the image point p from a perspective projection to a SOP of a point P_L located on the line of sight at the

same distance as P. P is on the line of sight L of p if, and only if, the image points p' and p'' are superposed. Then $\boldsymbol{cp'} = \boldsymbol{cp''}$, i.e. Eq. (3) is satisfied.

When we try to match the points P_k of an object to the lines of sight L_k of image points p_k, it is unlikely that all or even any of the points will fall on their corresponding lines of sight, or equivalently that $\boldsymbol{cp'_k} = \boldsymbol{cp''_k}$ or $\boldsymbol{p'_k p''_k} = \mathbf{0}$. We can minimize a global objective function E equal to the sum of the squared distances $d_k^2 = |\, \boldsymbol{p'_k p''_k}\, |^2$ between image points p'_k and p''_k:

$$E = \sum_k d_k^2 = \sum_k \left|\boldsymbol{cp'_k} - \boldsymbol{cp''_k}\right|^2 = \sum_k ((\mathbf{M}\cdot\mathbf{S}_k - w_k x_k)^2 + (\mathbf{N}\cdot\mathbf{S}_k - w_k y_k)^2) \quad (4)$$

where, to simplify the subsequent notation, we introduce the vectors (with four homogeneous coordinates) $\mathbf{S}_k = (\boldsymbol{P_0 P_k}, 1)$, and

$$\mathbf{M} = (M_1, M_2, M_3, M_4) = s(\mathbf{R}_1, T_x), \quad \mathbf{N} = (N_1, N_2, N_3, N_4) = s(\mathbf{R}_2, T_y).$$

We call $\mathbf{M}$ and $\mathbf{N}$ the *pose vectors*. In Fig. 1, notice that $\boldsymbol{p'p''} = s\boldsymbol{P'P''} = s\boldsymbol{PP_L}$. Thus this corresponds to minimizing the scaled sum of squared distances of model points along lines of sight, when distances are taken parallel to the image plane. This objective function is minimized iteratively. Initially, the w_k are all set to one. Then the following two operations take place at each step:

1. Compute the pose vectors $\mathbf{M}$ and $\mathbf{N}$ assuming w_k are known (Eq. (4)).
2. Compute the correction terms w_k using the $\mathbf{M}$ and $\mathbf{N}$ just computed (Eq. (1) for w)).

We now focus on the pose vectors $\mathbf{M}$ and $\mathbf{N}$. The objective function is minimized when the partial derivatives of E with respect to the coordinates of the pose vectors vanish. This condition provides 4×4 linear systems for $\mathbf{M}$ and $\mathbf{N}$ with solutions

$$\mathbf{M} = (\sum_k \mathbf{S}_k \mathbf{S}_k^T)^{-1} (\sum_k w_k x_k \mathbf{S}_k), \qquad \mathbf{N} = (\sum_k \mathbf{S}_k \mathbf{S}_k^T)^{-1} (\sum_k w_k y_k \mathbf{S}_k). \quad (5)$$

The matrix $L = (\sum_k \mathbf{S}_k \mathbf{S}_k^T)$ is a 4×4 matrix that can be precomputed.

4 Pose Calculation with Unknown Correspondences

The point p'' can be viewed as the image point p "corrected" for SOP using w computed at the previous step. The next step finds the pose such that the SOP of each point P is as close as possible to its corrected image point. Now when correspondences are unknown, each image point p_j can match any of the model points P_k, and must be corrected using the w corresponding to P_k:

$$w_k = \mathbf{R}_3 \cdot \boldsymbol{P_0 P_k}/T_z + 1. \quad (6)$$

Therefore for each image and model point p_j and P_k we generate a corrected image point p''_{jk}, aligned with the image center c and with p_j, and defined by

$$\boldsymbol{cp''_{jk}} = w_k \boldsymbol{cp_j}. \quad (7)$$

The scaled orthographic projections p'_k of the points P_k are

$$\boldsymbol{cp'_k} = \begin{bmatrix} \mathbf{M}\cdot\mathbf{S}_k \\ \mathbf{N}\cdot\mathbf{S}_k \end{bmatrix}. \quad (8)$$

The squared distances between the corrected points p''_{jk} and the SOPs are

$$d_{jk}^2 = \left|\boldsymbol{p'_k p''_k}\right|^2 = (\mathbf{M} \cdot \mathbf{S}_k - w_k x_j)^2 + (\mathbf{N} \cdot \mathbf{S}_k - w_k y_j)^2. \quad (9)$$

The simultaneous pose and correspondence problem can then be formulated as minimization of the global objective function

$$E = \sum_{j=1}^{J} \sum_{k=1}^{K} m_{jk}\, d_{jk}^2 = \sum_{j=1}^{J} \sum_{k=1}^{K} m_{jk} \left((\mathbf{M} \cdot \mathbf{S}_k - w_k x_j)^2 + (\mathbf{N} \cdot \mathbf{S}_k - w_k y_j)^2 \right) \quad (10)$$

where the m_{jk} are weights, equal to zero or one, for each of the d_{jk}^2, and J and K are the number of image and model points, respectively. The m_{jk} are correspondence variables that define the assignments between image and model feature points. Note that when all the assignments are well-defined, this objective function becomes equivalent to the objective function defined in Eq. (4).

This function E is minimized iteratively, as follows:

1. Compute the correspondence variables assuming everything else is fixed (see below).
2. Compute the pose vectors **M** and **N** assuming everything else is fixed (see below).
3. Compute the corrections w_k using the computed **M** and **N** (described above).

4.1 Pose Problem

We now focus on finding the optimal poses **M** and **N**, assuming the correspondence variables m_{jk} are known and fixed. As before, the minimizing pose vectors of E at a given step are those for which the partial derivatives of E with respect to these vectors vanish. This condition provides 4×4 linear systems for the coordinates of **M** and **N** whose solutions are

$$[\mathbf{M}, \mathbf{N}] = \left(\sum_{k=1}^{K} m'_k \mathbf{S}_k \mathbf{S}_k^{\mathsf{T}} \right)^{-1} \left[\sum_{j=1}^{J} \sum_{k=1}^{K} (m_{jk} w_k x_j \mathbf{S}_k), \sum_{j=1}^{J} \sum_{k=1}^{K} (m_{jk} w_k y_j \mathbf{S}_k) \right], \quad (11)$$

with $m'_k = \sum_{j=1}^{J} m_{jk}$. The terms $\mathbf{S}_k \mathbf{S}_k^{\mathsf{T}}$ are 4×4 matrices. Computing **M** and **N** requires inversion of a 4×4 matrix, $L = (\sum_{k=1}^{K} m'_k \mathbf{S}_k \mathbf{S}_k^{\mathsf{T}})$, which is inexpensive.

4.2 Correspondence Problem

We obtain the correspondence variables m_{jk} assuming that the d_{jk}^2 are known and fixed, and minimizing E. Our aim is to find a zero-one *assignment matrix*, $M = \{m_{jk}\}$, that specifies the matchings between a set of J image points and a set of K model points, and that minimizes E. This assignment matrix M has one row for each of the J image points p_j and one column for each of the K model points P_k. M must satisfy the constraint that each image point match at most one model point, and vice versa. A *slack row* $J + 1$ and a *slack column* $K + 1$ are added for points with no correspondences. A one in the slack column $K + 1$ at row j indicates that the image point p_j has no match among the model points. A one in the slack row $J + 1$ at column k indicates that the feature point P_k is not seen in the image. E will be a minimum if the assignment matrix M matches image and model points with the smallest distances d_{jk}^2. This problem is

solved by the iterative SoftAssign technique [19, 20]. We begin with a matrix M_0 in which element m^0_{jk} is initialized to $\exp(-\beta(d^2_{jk} - \alpha))$, with β very small, and with all slack elements set to a small constant. The parameter α determines how far apart two points must be before being considered unmatchable (see [20]). The continuous match matrix M_0 converges to a discrete matrix M using two concurrent procedures:

1. Each row and column of the correspondence matrix is normalized, alternately, by the sum of its elements. The resulting matrix then has positive elements with all *rows and columns summing to one*. (see Sinkhorn [34])
2. The term β is increased as the iteration proceeds. As β increases and each row or column of M_0 is renormalized, the terms m^0_{jk} corresponding to the smallest d^2_{jk} tend to converge to one while other elements tend to converge to zero. This is a deterministic annealing process [18] known as *Softmax* [4]. This is a desirable behavior, since it leads to an assignment of correspondence to matches that satisfy the matching constraints and whose sum of distances is minimized.

This procedure was called SoftAssign in [19, 20]. The resulting M is the assignment that minimizes E. This procedure along with the substeps that optimize pose and correct image points by SOP are combined into the iteration loop of SoftPOSIT.

4.3 Pseudocode for SoftPOSIT

The SoftPOSIT algorithm can be summarized as follows:

Inputs:

1. A list of J image feature points $p_j = (x_j, y_j)$.
2. A list of K world points $\mathbf{S}_k = (X_k, Y_k, Z_k, 1) = (\boldsymbol{P_0 P_k}, 1)$ in the object.

Initialize slack elements of assignment matrix M to $\gamma = 1/(\max\{J, K\} + 1)$, β to β_0 ($\beta_0 \approx 4 \times 10^{-4}$ if the pose is unconstrained, larger if a good initial guess is available).
Initialize pose vectors **M** and **N** with expected pose or a random pose.
Initialize $w_k = 1$.
Do A until $\beta > \beta_{final}$ (β_{final} around 0.5) (*Deterministic annealing loop*)

- Compute squared distances $d^2_{jk} = (\mathbf{M} \cdot \mathbf{S}_k - w_k x_j)^2 + (\mathbf{N} \cdot \mathbf{S}_k - w_k y_j)^2$
- Compute $m^0_{jk} = \gamma \exp(-\beta(d^2_{jk} - \alpha))$
- **Do B until** ΔM **small** (*Sinkhorn's method*)
  - Update matrix M by normalizing across all rows: $m^1_{jk} = m^0_{jk} / \sum_{k=1}^{K+1} m^0_{jk}$
  - Update matrix M by normalizing across all columns: $m^0_{jk} = m^1_{jk} / \sum_{j=1}^{J+1} m^1_{jk}$
- **End Do B**
- Compute 4×4 matrix $L = (\sum_{k=1}^{K} m'_k \mathbf{S}_k \mathbf{S}_k^{\mathsf{T}})$ with $m'_k = \sum_{j=1}^{J} m_{jk}$
- Compute L^{-1}
- Compute $\mathbf{M} = L^{-1}(\sum_{j=1}^{J} \sum_{k=1}^{K} m_{jk} w_k \, x_j \mathbf{S}_k)$
- Compute $\mathbf{N} = L^{-1}(\sum_{j=1}^{J} \sum_{k=1}^{K} m_{jk} w_k \, y_j \mathbf{S}_k)$
- Compute $s = |(M_1,\ M_2,\ M_3)|$, $\mathbf{R}_1 = (M_1, M_2, M_3)/s$, $\mathbf{R}_2 = (N_1, N_2, N_3)/s$, $\mathbf{R}_3 = \mathbf{R}_1 \times \mathbf{R}_2$
- Compute $w_k = \mathbf{R}_3 \cdot \boldsymbol{P_0 P_k} / T_z + 1$
- $\beta = \beta_{update} \beta$ (β_{update} around 1.05)

End Do A

Outputs: Rotation matrix $R = [\mathbf{R}_1 \ \mathbf{R}_2 \ \mathbf{R}_3]^{\mathsf{T}}$, translation vector $\mathbf{T} = (T_x, T_y, T_z)$, and assignment matrix $M = \{m_{jk}\}$ between image and world points.

5 Random Start SoftPOSIT

The SoftPOSIT algorithm described above performs a deterministic annealing search starting from an initial guess for the object's pose. There is no guarantee of finding the global optimum. The probability of finding the globally optimal object pose and correspondences starting from an initial guess depends on a number of factors including the number of model points, the number of image points, the number of occluded model points, the amount of clutter in the image, and the image measurement noise. A common way of searching for a global optimum, and the one taken here, is to run the algorithm starting from a number of different initial guesses, and keep the first solution that meets a specified termination criteria. Our initial guesses range over $[-\pi, \pi]$ for the three Euler angles, and over a 3D space of translations containing the true translation.

Since each search for correspondence and pose is relatively expensive, we would like to have a mathematical statement that allows us to make the claim that, for a given number of starting points, our starting guesses sample the parameter space in some optimal manner. Fortunately, there are a set of deterministic points that have such properties. These are the quasi-random, or low discrepancy sequences [30]. Unlike points obtained from a standard pseudo-random generator, quasi-random points are optimally self-avoiding, and uniformly space filling. We use a standard quasi-random generator [10] to generate quasi-random 6-vectors in a unit 6D hypercube. These points are scaled to reflect the expected ranges of translation and rotation.

5.1 Search Termination

Ideally, one would like to repeat the search from a new starting point whenever the number of correspondences determined is not maximal. However, one usually does not know what this maximal number is. Instead, we repeat the search when the number of model points matching to image points is less than some threshold t_m. Due to occlusion and imperfect image feature extraction, not all model points will be detected as features in an image. Let p_d be the ratio of the number of model points detected as image features to the total number of model points ($p_d \in [0, 1]$). In the Monte Carlo simulations described below, p_d is known. With real imagery, however, p_d must be estimated based on the scene complexity and the reliability of the feature detection algorithm used.

We terminate the search for better solutions when the current solution is such that the number of model points matching to image points is greater than or equal to the threshold $t_m = \rho p_d K$ where $\rho \in [0, 1]$ determines the fraction of model points to be matched and K is the total number of model points. ρ accounts for measurement noise. In the experiments discussed below, we take $\rho = 0.8$. This test is not perfect, as it is possible for a pose to be very accurate even when the number of matched points is less than this threshold; this occurs mainly in cases of high noise. Conversely, a wrong pose may be accepted when the ratio of clutter features to detected model points is high. However, these situations are uncommon.

5.2 Early Search Termination

The deterministic annealing loop of the SoftPOSIT algorithm iterates over a range of values for the annealing parameter β_k. In the experiments reported here, β_0 is initialized to 0.0004 and is updated according to $\beta_{k+1} = 1.05 \times \beta_k$, and the iteration ends when $\beta_k > 0.5$, or earlier if convergence is detected. This means that the annealing loop can

run for up to 147 iterations. It's usually the case that, by viewing the original image and, overlaid on top of this, the projected model points produced by SoftPOSIT, a person can determine early on in the iteration (e.g., around iteration 30) whether or not the algorithm is going to converge to the correct pose. It is desired that the algorithm make this determination itself, so that it can end the current unfruitful search and restart.

A simple test is performed on *each* iteration to determine if it should continue or restart. At iteration k of SoftPOSIT, the match matrix $M^k = \{m_{i,j}^k\}$ is used to predict the final correspondences of model to image points: upon convergence we expect image point i to correspond to model point j if $m_{i,j}^k > m_{u,v}^k$ for all $u \neq i$ and all $v \neq j$ (however, this is not guaranteed). The number of predicted correspondences at iteration k, n_k, is the number of pairs (i, j) that satisfy this relation. We define the match ratio on iteration k as $r_k = n_k/(p_d K)$ where p_d is defined above. This metric is commonly used at the end of a local search to determine if the current solution for correspondence and pose is good enough to end the search for the global optimum. We, however, use this metric within the local search itself. Let *CP* denote the event that the SoftPOSIT algorithm eventually converges to the correct pose. Then, the algorithm restarts after the k^{th} iteration if $P(CP \mid r_k) < \alpha P(CP)$ where $0 < \alpha \leq 1$. That is, the search is restarted from a new random starting condition whenever the posterior probability of eventually finding a correct pose given r_k drops to less than some fraction of the prior probability of finding the correct pose. A separate posterior probability is required for each k because the ability to predict the outcome using r_k improves as the iteration progresses. Although this test may result in termination of some searches which would eventually produce a good pose, it is expected on average that the total time required to find a good pose will be less. Our experiments show that this is true; we obtain a speedup by a factor of at least two.

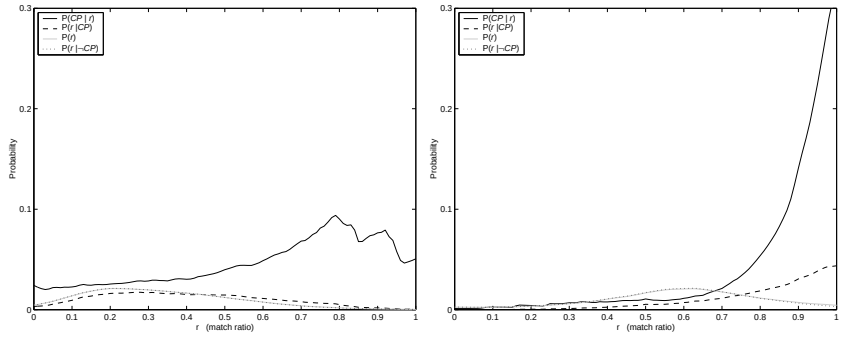
The posterior probability function for the k^{th} iteration can be computed from $P(CP)$, the prior probability of finding a correct pose on one random local search, and from $P(r_k \mid CP)$ and $P(r_k \mid \overline{CP})$, the probabilities of observing a particular match ratio on the k^{th} iteration given that the eventual pose is either correct or incorrect, respectively:

$$P(CP \mid r_k) = \frac{P(CP)P(r_k \mid CP)}{P(CP)P(r_k \mid CP) + P(\overline{CP})P(r_k \mid \overline{CP})}.$$

$P(CP)$, $P(\overline{CP})$, $P(r_k \mid CP)$, and $P(r_k \mid \overline{CP})$ are estimated in Monte Carlo simulations of the algorithm in which the number of model vertices and the levels of image clutter, occlusion, and noise are all varied. To estimate $P(r_k \mid CP)$ and $P(r_k \mid \overline{CP})$, the algorithm is repeatedly run on random test data. For each test, the values of the match ratio r_k computed at each iteration are recorded. Once an iteration completes, ground truth information is used to determine whether or not the correct pose was found. If the pose is correct, then the recorded values of r_k are used to update histograms representing $P(r_k \mid CP)$; otherwise, histograms for $P(r_k \mid \overline{CP})$ are updated. Upon completing the training, the histograms are normalized. See Fig. 2.

6 Experiments

We investigate two important questions related to the performance of the SoftPOSIT algorithm: (a) How often does it find a "good" pose? (b) How long does it take?



(a) (b)

Fig. 2. Probability functions estimated for (a) the first iteration, and (b) the 31st iteration, of the SoftPOSIT algorithm.

6.1 Synthetic Data

The algorithm has been extensively evaluated in Monte Carlo simulations. The simulations are characterized by 5 parameters: n_t, K, p_d, p_c, and σ. The parameter n_t is the number of trials performed for each combination of values for the remaining 4 parameters. K is the number of points in a 3D model. p_d is the probability that the image of any particular model point will be detected. p_c is the probability that any particular image point is clutter, i.e., is not the image of some 3D model point. Finally, σ is the standard deviation of the normally distributed noise in the image coordinates of the non-clutter feature points, measured in pixels for a 1000×1000 image, generated by a simulated camera having a 37 degree field of view (focal length of 1500 pixels). The current tests were performed with $n_t = 100$, $K \in \{20, 30, 40, 50\}$, $p_d \in \{0.4, 0.6, 0.8\}$, $p_c \in \{0.2, 0.4, 0.6\}$, and $\sigma \in \{0.5, 1.0, 2.5\}$. There were 10800 independent trials.

For each trial, a 3D model is created in which the K model vertices are randomly located, with uniform probability, in a sphere centered at the origin. Because the algorithm works with points, only the model vertices are important. However, to make human interpretation of results easier, each model vertex is connected to the two closest remaining model vertices. The model is placed with random rotation and translation in the camera view. Each projected model point is detected with probability p_d. We add Gaussian noise ($\mathcal{N}(0, \sigma)$) to both x and y image coordinates. Finally, $Kp_d/(1-p_c)$ randomly located clutter feature points are added to the true feature points, so that $100 \times p_c$ % of the feature points are clutter. Fig. 3 shows cluttered images of random models.

We consider a pose to be *good* when it allows 80% ($\rho = 0.8$ in section 5.1) or more of the detected model points to be matched to image points. The number of random starts for each trial was limited to 10^4. If a good pose is not found after 10^4 starts, the algorithm declares failure. Fig. 4 shows two examples of the pose found.

Fig. 5 shows the success rate as a function of the number of model points for the case of $\sigma = 2.5$ and for all combinations of the parameters p_d and p_c. (Due to space

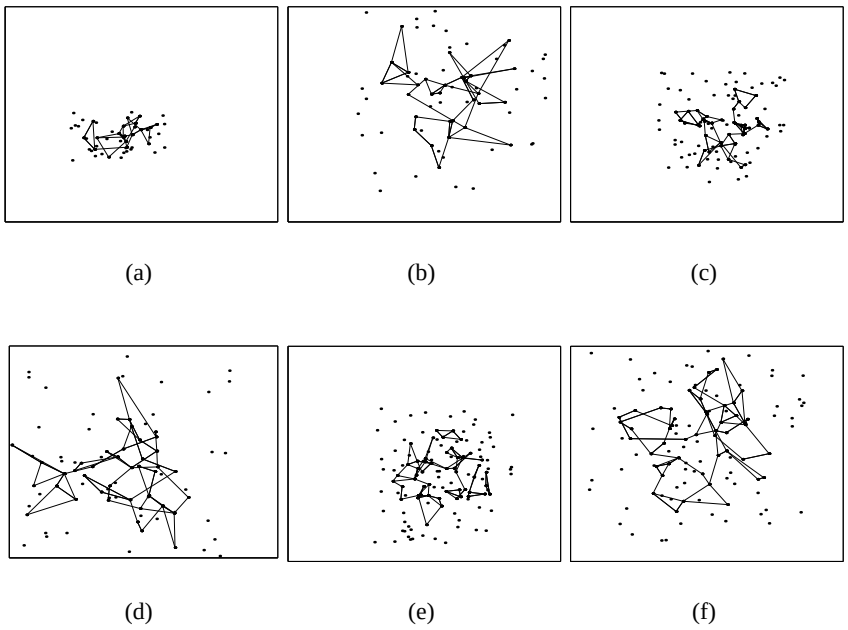
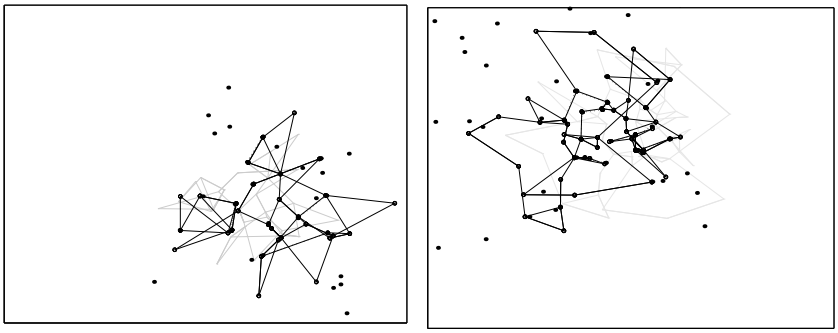


Fig. 3. Typical images of randomly generated models. The number of model points in the models are 20 – (a), 30 – (b), 40 – (c) and (d), and 50 – (e) and (f). In all cases $p_d = 1.0$ and $p_c = 0.6$.

limitations, we only describe results for the case of $\sigma = 2.5$; the algorithm performs better for smaller σ's.) For more than 86% of the different combinations of simulation parameters, a good pose is found in 95% or more of the associated trials. For the remaining 14% of the tests, a good pose is found in 85% or more of the trials. (The overall success rate is 94%.) As expected, the higher the occlusion rate (lower p_d) and the clutter rate (higher p_c), the lower the success rate. For the high-clutter and high-occlusion tests, the success rate increases as the number of model points decreases. This is because a smaller number of model points are more easily matched to clutter than a larger number of model points. Fig. 6 shows the average number of random starts required to find a good pose. These numbers generally increase with increasing image clutter and occlusion.

The run-time complexity of SoftPOSIT (for a single start) is easily seen to be $O(JK)$ where J is the number of image points and K is the number of model points. Our results show that the mean number of random starts required to find a good pose, to ensure a probability of success of at least 0.95 in all but the highest clutter and occlusion levels, is bound by a function that is linear in the size of the input. That is, the mean number of random starts is $O(J)$, assuming that $K < J$, as is normally the case. Then, the run-time complexity of SoftPOSIT *with* random starts is $O(KJ^2)$. This is better than any known algorithm that solves simultaneous pose and correspondence problem under full perspective.



(a) (b)

Fig. 4. Two cluttered images with projected models overlayed for which we found a good pose. The black dots are the original image points; white dots are projections of model points for the computed pose, and the gray lines are the initial guess that lead to the true pose being found.

6.2 Experiments with Images

We applied the SoftPOSIT algorithm to imagery generated from a model of a district of Los Angeles by a commercial virtual reality (VR) system. Fig. 7 shows an image generated and a world model projected into that image using the pose computed by SoftPOSIT. Image feature points are automatically located in the image by detecting corners along the boundary of bright sky regions. Because the 3D world model has over 100,000 data points, we use a rough pose estimate (as may be generated by a GPS system) to cull the majority of model points that are far from the estimated field of view. The world points that do fall into this estimated view are further culled by keeping only those that project near the detected skyline. So far, the results have been very good, and could be used for autonomous navigation. Although this is not real imagery, the VR system used is advanced, and should give a good indication of how the system will perform on real imagery.

7 Conclusions

We have developed and tested the SoftPOSIT algorithm for determining the correspondence and pose of objects in an image. This algorithm will be used as a component in an object recognition system. Our evaluation indicates that the algorithm performs well under a variety of levels of occlusion, clutter, and noise. We are currently collecting imagery and 3D models of city environments for the purpose of evaluating the algorithm on real data.

The complexity of SoftPOSIT has been empirically determined to be $O(KJ^2)$. This is better than any known algorithm that solves the simultaneous pose and correspondence problem for a full perspective camera model. Rigorous validation of this claim is an item of future work.

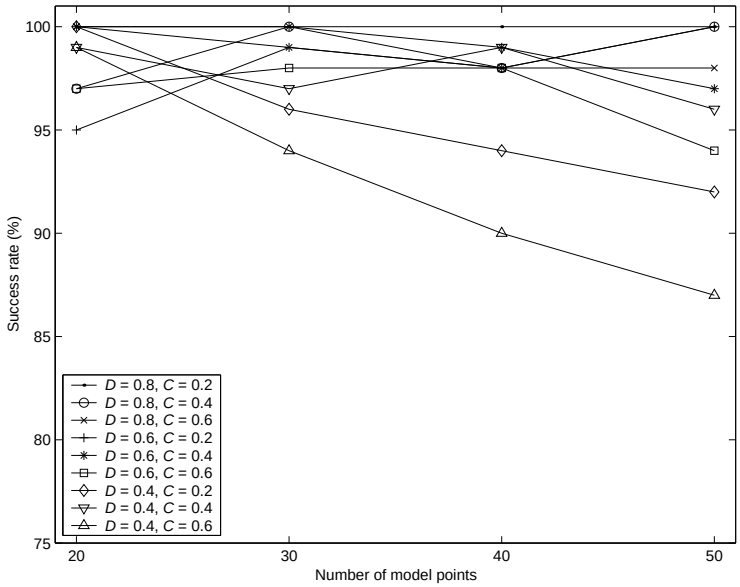
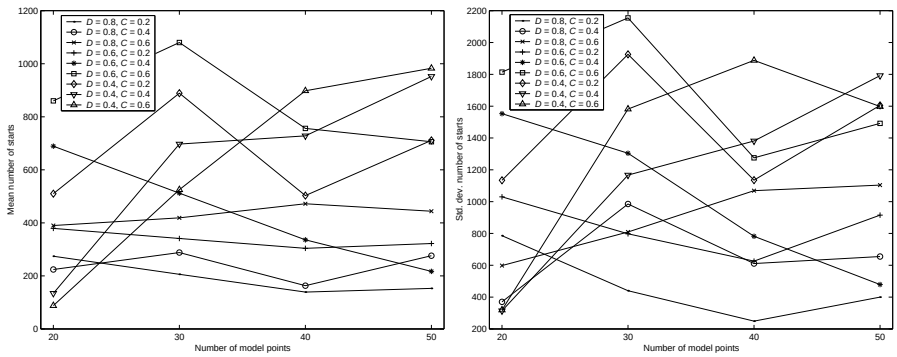


Fig. 5. Success rate as a function of the number of model points for fixed values of p_d and p_c. (Note that p_d and p_c are denoted D and C, respectively, in the legend of this figure and in the next few figures.)

References

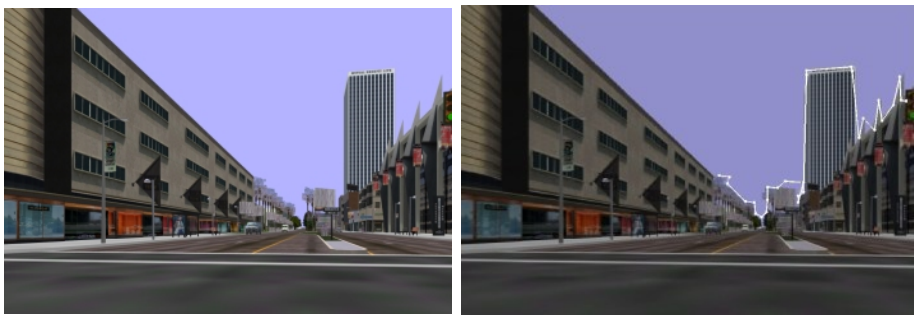
1. H.S. Baird. *Model-Based Image Matching Using Location*, MIT Press, 1985.
2. J.S. Beis & D.G. Lowe, "Indexing Without Invariants in 3D Object Recognition," *IEEE Trans. PAMI*, vol. 21, no. 10, pp. 1000–1015, 1999.
3. J.R. Beveridge & E.M. Riseman, "Optimal Geometric Model Matching Under Full 3D Perspective," *CVIU*, vol. 61, pp. 351–364, May 1995.
4. J.S. Bridle. "Training Stochastic Model Recognition as Networks can Lead to Maximum Mutual Information Estimation of Parameters", *Advances in Neural Information Processing Systems*, vol. 2, pp. 211–217, 1990.
5. J.B. Burns, R.S. Weiss, & E.M. Riseman, "View Variation of Point-Set and Line-Segment Features," *IEEE Trans. PAMI*, vol. 15, pp. 51–68, 1993.
6. T.M. Breuel. "Fast Recognition using Adaptive Subdivisions of Transformation Space," *Proc. 1992 IEEE CVPR*, pp. 445–451.
7. T.A. Cass. "Polynomial-Time Object Recognition in the Presense of Clutter, Occlusion, and Uncertainty," *Proc. ECCV'92*, pp. 834–842, Springer-Verlag, 1992.
8. T.A. Cass. "Robust Geometric Matching for 3D Object Recognition," *Proc. 12th ICPR*, vol. 1, pp. 477 –482, 1994.
9. T.A. Cass. "Robust Affine Structure Matching for 3D Object Recognition," *IEEE Trans. PAMI*, vol. 20, no. 11, pp. 1265–1274, 1998.
10. "C" code for quasi-random number generation. Available from http://www.taygeta.com.
11. P. David, D. DeMenthon, R. Duraiswami, & H. Samet, "Evaluation of the SoftPOSIT Model to Image Registration Algorithm," University of Maryland Technical Report CAR-TR-974, February 2002.



(a) (b)

Fig. 6. Number of random starts required to find a good pose as a function of the number of model points for fixed values of p_d and p_c. (a) Mean . (b) Standard deviation.

12. D. DeMenthon & L.S. Davis. "Recognition and Tracking of 3D Objects by 1D Search," *Proc. DARPA Image Understanding Workshop*, Washington, DC, April 1993.
13. D. DeMenthon & L.S. Davis, "Model-Based Object Pose in 25 Lines of Code", *International Journal of Computer Vision*, vol. 15, pp. 123–141, 1995.
14. D. DeMenthon & P. David, "SoftPOSIT: An Algorithm for Registration of 3D Models to Noisy Perspective Images Combining SoftAssign and POSIT," University of Maryland Technical Report CAR-TR-970, May 2001.
15. R.W. Ely, J.A. Digirolamo, & J.C. Lundgren, "Model Supported Positioning," *Integrating Photogrammetric Techniques with Scene Analysis and Machine Vision II, SPIE Aerospace Sensing and Dual Use Sensors and Controls*, Orlando, April 1995.
16. P.D. Fiore. "Efficient Linear Solution of Exterior Orientation," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 23, no. 2 , pp. 140–148, February 2001.
17. M.A. Fischler & R.C. Bolles, "Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography," *Comm. Association for Computing Machinery*, vol. 24, no. 6, pp. 381–395, June 1981.
18. D. Geiger & A.L. Yuille, "A Common Framework for Image Segmentation", *International Journal of Computer Vision*, vol. 6, pp. 227–243, 1991.
19. S. Gold & A. Rangarajan, "A Graduated Assignment Algorithm for Graph Matching", *IEEE Trans. PAMI*, vol. 18, pp. 377–388, 1996.
20. S. Gold, A. Rangarajan, C.P. Lu, S. Pappu, and E. Mjolsness, "New Algorithms for 2D and 3D Point Matching: Pose Estimation and Correspondence", *Pattern Recognition*, vol. 31, pp. 1019–1031, 1998.
21. E. Grimson, *Object Recognition by Computer: The Role of Geometric Constraint*, MIT Press, 1990.
22. E. Grimson & D.P. Huttenlocher, "On the Verification of Hypothesized Matches in Model-Based Recognition", *IEEE Trans. PAMI*, vol. 13, no. 12, pp. 1201–1213, 1991.
23. R. Hartley & A. Zisserman. *Multiple View Geometry in Computer Vision.* Cambridge University Press, 2000.
24. R. Horaud, B. Conio, O. Leboulleux, & B. Lacolle, "An Analytic Solution for the Perspective 4-Point Problem," *Proc. 1989 IEEE CVPR*, pp. 500–507.



(a) (b)

Fig. 7. (a) Original image from a virtual reality system. (b) World model (white lines) projected into this image using the pose computed by SoftPOSIT.

25. B.K.P. Horn. *Robot Vision*. MIT Press, Cambridge, Massachusetts 1986.
26. D.W. Jacobs. "Space Efficient 3D Model Indexing," *Proc. 1992 IEEE CVPR*, pp. 439–444.
27. F. Jurie, "Solution of the Simultaneous Pose and Correspondence Problem Using Gaussian Error Model," *CVIU*, vol. 73, pp. 357–373, 1999.
28. Y. Lamdan & H.J. Wolfson. "Geometric Hashing: A General and Efficient Model-Based Recognition Scheme," *Proc. 1998 IEEE ICCV*, pp. 238-249.
29. C.-P. Lu, G.D. Hager, & E. Mjolsness, "Fast and Globally Convergent Pose Estimation from Video Images," *IEEE Trans. PAMI*, vol. 22, pp. 610–622, 2000.
30. W.J. Morokoff & R. E. Caflisch, "Quasi-Random Sequences and their Discrepancies", *SIAM J. Sci. Comput.*, pp. 1251–1279, 1994.
31. H. Murase & S.K. Nayar, "Visual Learning and Recognition of 3-D Objects from Appearance," *IJCV*, vol. 14, pp. 5–24, 1995.
32. C.F. Olson, "Efficient Pose Clustering Using a Randomized Algorithm," *IJCV*, vol. 23, no. 2, pp. 131–147, 1997.
33. S. Procter & J. Illingworth. "ForeSight: Fast Object Recognition using Geometric Hashing with Edge-Triple Features," *Proc. 1997 ICIP.*, vol. 1, pp. 889–892.
34. R. Sinkhorn. "A Relationship between Arbitrary Positive Matrices and Doubly Stochastic Matrices", *Annals Math. Statist.*, vol. 35, pp. 876–879, 1964.
35. S. Ullman, "Aligning Pictorial Descriptions: An Approach to Object Recognition," *Cognition*, vol. 32, no. 3, pp. 193–254, 1989.
36. P. Wunsch & G. Hirzinger, "Registration of CAD Models to Images by Iterative Inverse Perspective Matching", *Proc. 1996 ICPR*, pp. 78–83.
37. J.-C. Yuan, "A General Photogrammetric Method for Determining Object Position and Orientation," *IEEE Trans. Robotics and Automation*, vol. 5, no. 2, pp. 129-142, 1989.