
Georg F. Mauer
University of Nevada, Las Vegas, georg.mauer@unlv.edu

Follow this and additional works at: https://digitalscholarship.unlv.edu/hrc_trp_fuels

Part of the Nuclear Engineering Commons, and the Robotics Commons

Repository Citation
Available at: https://digitalscholarship.unlv.edu/hrc_trp_fuels/25

This Report is brought to you for free and open access by the Transmutation Research Program Projects at Digital Scholarship@UNLV. It has been accepted for inclusion in Fuels Campaign (TRP) by an authorized administrator of Digital Scholarship@UNLV. For more information, please contact digitalscholarship@unlv.edu.
Design and Evaluation of Processes for Fuel Fabrication

QUARTERLY PROGRESS REPORT #5

UNLV AAA University Participation Program

Prepared by: Georg F. Mauer
Department of Mechanical Engineering
UNLV, Las Vegas, NV 89154-4027
Phone: (702) 895-3830
FAX : (702) 895-3936

Reporting Period: September 1, 2002 through November 30, 2002
Design and Evaluation of Processes for Fuel Fabrication

Summary

The fifth quarter of the project covered the following:

- Literature Search: The process of evaluating the pertinent literature continued.
- Mr. Richard Silva continued the development of a simulation model with a Waelischmiller hot cell robot. Rich will continue to develop detailed 3-D process simulation models as his M.Sc. thesis project. Rich is employed with Bechtel at the Yucca Mountain project.
- Concepts and Methods for Vision-Based Hot Cell Supervision and control (Ph.D. Student Jae-Kyu Lee)
- Dr. Mauer presented a paper at the ISCA 2002 conference in San Diego, CA titled: “Object Recognition Over An Expanded Range Of Viewing Angles Using Indexing Methods

Part I Hot Cell Manipulator Simulation

As previously reported, graduate student Richard Silva developed a 3D simulation model for a Waelischmiller-type hot cell robot (see Fig. 1). During the reporting period, we began the development of a simulation model for robot control, using Matlab. Matlab is capable of interfacing with the spatial robot model. Please note that the robot model comprises a geometric model as well as the modeling of the robot dynamics. Thus a realistic simulation of the forces and torques present during robot motion can be generated. Furthermore, the
combination of Matlab with Visual Nastran 4D will permit the detailed simulation of events such as pick and place operations as well as unusual events such as collisions and impacts.

The following two pages of Mathcad analysis illustrate the mathematical modeling process for the robot kinematics. The inverse kinematics problem (finding the joint angles to achieve a desired end effector position) cannot generally solved in closed form due to the requirement to invert a nonlinear matrix. In addition, the solution degenerates at various points (e.g. when either sine or cosine terms in the denominator approach zero.) Lastly, a large portion of the work space can be reached with multiple arm configurations (see Fig. 2). The typical approach to the inverse kinematics problem is linearization of the joint matrix T (Jacobian matrix) about the current operating point. The Jacobian can be inverted in many cases. Otherwise iterative methods must be applied.

Literature Search – A large body of publications pertaining to robot kinematics, path planning, and control exists. A selection is listed at the end of this report.

The program development for robot control was begun using Matlab and the Matlab robotics toolbox. Figures 3 and 4 illustrate the approach on the example of a Puma-type robot (source: Matlab robotics toolbox).
Kinematic Analysis of the Wälischmiller Manipulator

The input angles for the joints (β) are as follows:

Joint 2 := 0 deg  Joint 3 := 0 deg  Joint 5 := 0 deg  Joint 6 := 0 deg  Joint 7 := 0 deg

There are a total of 5 joints where it takes 11 coordinate transformations to correctly model the system.

<table>
<thead>
<tr>
<th>θ</th>
<th>α</th>
<th>a</th>
<th>d</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>24</td>
</tr>
<tr>
<td>Joint 2 := 90 deg</td>
<td>0</td>
<td>42.5</td>
<td></td>
</tr>
<tr>
<td>Joint 3 := −90 deg</td>
<td>0</td>
<td>10.5</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>90 deg</td>
<td>0</td>
<td>41.5 + 9</td>
</tr>
<tr>
<td>Joint 5 := −90 deg</td>
<td>0</td>
<td>10.5</td>
<td></td>
</tr>
<tr>
<td>Joint 6 := 90 deg</td>
<td>0</td>
<td>29.5</td>
<td></td>
</tr>
<tr>
<td>Joint 7 := 90 deg</td>
<td>0</td>
<td>13.0</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>90 deg</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>−90 deg</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>−90 deg</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Key dimensions for the manipulator are as follows:

θ := dh(1)
α := dh(2)
a := dh(3)
d := dh(4)

General Form of the Rotational Joint Matrix

\[
\begin{bmatrix}
cos(\theta_1) & -cos(\alpha_1)sin(\theta_1) & sin(\alpha_1)sin(\theta_1) & a_1cos(\theta_1) \\
sin(\theta_1) & cos(\alpha_1)sin(\theta_1) & -sin(\alpha_1)cos(\theta_1) & a_1sin(\theta_1) \\
0 & sin(\alpha_1) & cos(\alpha_1) & d_1 \\
0 & 0 & 0 & 1
\end{bmatrix}
\]

All joints are rotational, and therefore, the general form (A_i) of the rotational joint matrix will be used. Solving for the General Rotational Matrices:

\[
A_i := \begin{bmatrix}
cos(\theta_i) & -cos(\alpha_i)sin(\theta_i) & sin(\alpha_i)sin(\theta_i) & a_i cosm(\theta_i) \\
sin(\theta_i) & cos(\alpha_i)sin(\theta_i) & -sin(\alpha_i)cos(\theta_i) & a_i sin(\theta_i) \\
0 & sin(\alpha_i) & cos(\alpha_i) & d_i \\
0 & 0 & 0 & 1
\end{bmatrix}
\]

The homogeneous matrix (T) specifies the location of the ith coordinate frame with respect to the base coordinate system. T is the chain product of successive coordinate transformation matrices of A, and is expressed as:
\[ T := A_1 \cdot A_2 \cdot A_3 \cdot A_4 \cdot A_5 \cdot A_6 \cdot A_7 \cdot A_8 \cdot A_9 \cdot A_{10} \cdot A_{11} \]

Or, written in the more general form:

\[ T := \prod_{j=1}^{n} A_j \quad T = \begin{pmatrix}
    0 & 1 & 0 & 0 \\
    -1 & 0 & 0 & -21 \\
    0 & 0 & 1 & 164.25 \\
    0 & 0 & 0 & 1
\end{pmatrix} \]

Where, if expanded for each \( n \)th value:

\[ T_1 := \prod_{j=1}^{1} A_j \quad T_1 = \begin{pmatrix}
    1 & 0 & 0 & 0 \\
    0 & 1 & 0 & 0 \\
    0 & 0 & 1 & 24 \\
    0 & 0 & 0 & 1
\end{pmatrix} \]

\[ T_2 := \prod_{j=1}^{2} A_j \quad T_2 = \begin{pmatrix}
    1 & 0 & 0 & 0 \\
    0 & 0 & -1 & 0 \\
    0 & 1 & 0 & 66.5 \\
    0 & 0 & 0 & 1
\end{pmatrix} \]

\[ T_3 := \prod_{j=1}^{3} A_j \quad T_3 = \begin{pmatrix}
    1 & 0 & 0 & 0 \\
    0 & 1 & 0 & -10.5 \\
    0 & 0 & 1 & 66.5 \\
    0 & 0 & 0 & 1
\end{pmatrix} \]

\[ T_4 := \prod_{j=1}^{4} A_j \quad T_4 = \begin{pmatrix}
    1 & 0 & 0 & 0 \\
    0 & 0 & -1 & -10.5 \\
    0 & 1 & 0 & 117 \\
    0 & 0 & 0 & 1
\end{pmatrix} \]

\[ T_5 := \prod_{j=1}^{5} A_j \quad T_5 = \begin{pmatrix}
    1 & 0 & 0 & 0 \\
    0 & 1 & 0 & -21 \\
    0 & 0 & 1 & 117 \\
    0 & 0 & 0 & 1
\end{pmatrix} \]

\[ T_6 := \prod_{j=1}^{6} A_j \quad T_6 = \begin{pmatrix}
    1 & 0 & 0 & 0 \\
    0 & 0 & -1 & -21 \\
    0 & 1 & 0 & 146.5 \\
    0 & 0 & 0 & 1
\end{pmatrix} \]

\[ T_7 := \prod_{j=1}^{7} A_j \quad T_7 = \begin{pmatrix}
    0 & -1 & 0 & 0 \\
    0 & 0 & -1 & -21 \\
    1 & 0 & 0 & 151.25 \\
    0 & 0 & 0 & 1
\end{pmatrix} \]

\[ T_8 := \prod_{j=1}^{8} A_j \quad T_8 = \begin{pmatrix}
    0 & -1 & 0 & 0 \\
    0 & 0 & -1 & -21 \\
    1 & 0 & 0 & 164.25 \\
    0 & 0 & 0 & 1
\end{pmatrix} \]

\[ T_9 := \prod_{j=1}^{9} A_j \quad T_9 = \begin{pmatrix}
    0 & -1 & 0 & 0 \\
    0 & 1 & 0 & -21 \\
    1 & 0 & 0 & 164.25 \\
    0 & 0 & 0 & 1
\end{pmatrix} \]

\[ T_{10} := \prod_{j=1}^{10} A_j \quad T_{10} = \begin{pmatrix}
    0 & 0 & -1 & 0 \\
    -1 & 0 & 0 & -21 \\
    0 & 1 & 0 & 164.25 \\
    0 & 0 & 0 & 1
\end{pmatrix} \]
Part II Object Recognition

**Concept** – Objects are recognized from CCD camera images based on their geometrical features. Machine recognition is the matching of an image pattern with known patterns generated by the same object class and stored in a database. Since detected patterns vary considerably with distance, viewing angle, lighting conditions, and also occlusions, a systematic method for reducing and organizing the vast quantities of possible reference data would be helpful. Our approach identifies objects by their contours. The database stores only sets of characteristic points, such as edges and corners. Indexing employs a priori stored information about the object models, in order to quickly eliminate non-compatible model-scene feature matches during recognition. Hence, only the most feasible matches are considered, that is, the matches where the model features could have projected to the scene features. Indexing-based methods usually employ a hash scheme to efficiently store and retrieve information about the models into a hash table. During preprocessing, groups of model features are considered, and a description for each is computed and stored in the indexed location. During recognition, groups of scene points are used to access the hash table. An unknown object is identified using geometric interpolation between a finite set
of stored views. A more detailed discussion is contained in the appended CAINE 2002 conference paper. A nearest neighbor (NN) algorithm establishes the best match between candidate object and the stored views in the database. The recognition of objects in full (unoccluded) view requires as a minimum a match of three characteristic points with their reference in the database.

For complex objects with overlapping geometries, simple matching indexing vector triplets are not sufficient. We need also the related regional information connecting multiple vector triplets from the original image in order to make a correct decision (see Figure 5).

Figure 5: Comparison between raw indexing and regional indexing
(a) Raw indexing for pattern recognition. Here, indexing vectors are computed individually from base point P to each corner point.
(b) Target image
(c) Regional indexing for pattern recognition. Here, indexing vectors are computed from base points P and Q to corner points but also they are computed partially.

Fig. 5 shows the concept of regional index. Here we have a stapler image whose base is partly occluded. To identify the stapler’s hidden base, we must compute the regional (or partial) index vector d from point Q rather than P. We match each vector separately with the target image (stapler) and verify the correctness of the geometrical relation.

Pattern Matching Example - Consider the stapler in Fig. 5. To predict hidden base, we need clustered indexing vector containing information about base of stapler. Fig. 6 contains two clusters that represent rectangular shapes. Alignment with base point Q yields the best match. We also have a priori knowledge that clusters 1 and 2 are shape elements of the stapler’s base. While we did begin at point Q in this example, a real pattern match would have to examine all possible
point sets one at the time, making the search easily inefficient and slow if large data bases must be searched.

![Figure 6: Search for the Base](image)

Two clusters. Seeking to align the clusters with base point Q.

**(d) Test 1: alignment of cluster 1 (red)**  
**(e) Test 2: alignment of cluster 2 (green)**  
**(f) Final verification (blue)**

**Knowledge-Based Pattern Recognition** – We are presently developing algorithms for rule-based (or knowledge-based) pattern recognition. The rules will be used to connect separate vector triplets to the known geometries of the objects from which they were extracted. The basic concept of knowledge base (or rule base) follows:

**IF (a set of conditions is satisfied)**  
**THEN (a set of consequences can be executed)**

In the example of Fig. 6, we employ three rules.
Rule #1:
IF (either cluster 1 or cluster 2 is matching indexing vector in the test image)
THEN (store those region of index with those matching error)

Rule #2:
IF (amongst all matches passing through rule #1, their geometry matches rectangular shape)
THEN (store those cases with their vector ratio)

Rule #3:
IF (amongst all matches passing through rule #2, their error of vector ratio is smallest)
THEN (conclude correct match $\rightarrow$ stapler’s base)

The concept of nearest neighbor search is applied during every decision step. The decision tree flow chart of this procedure is shown in Figure 7.

**Figure 7:** Decision Tree in Nearest Neighbor Search
Each circle represents a regional indexing data point.
Each branch represents an applied rule.
Graduate Student Progress

Ph.D. student Jae-Kyu Lee presented his dissertation proposal to the doctoral advisory committee and passed the preliminary examination. He is making good progress towards delivering a practical machine recognition algorithm that can be used for reliable automated recognition and visual process supervision in the hot cell.

Conclusion

During the fifth quarter, project needs and issues were detailed further. Dr. Mauer’s visits to four laboratories and plants provided insight into current practices and ongoing R&D. The present effort is focusing on crating detailed manufacturing simulation models, and on collecting more detailed information on the cost and space requirements for powder processing equipment. Matlab software, another essential part of the plant simulation was received. Matlab will provide a control interface for the work cell simulation. We expect the simulations to become more realistic with the addition of Matlab.

Management Issues: Expenditures were generally as planned in the proposal. Ph. D. student Jae-Kyu Lee was funded throughout the reporting period. An undergraduate engineering student, Mr. Timothy Atobatele, has been retained on an hourly basis to provide support services. A second undergraduate student in Mechanical Engineering, Mr. Jamil Renno, has been retained on an hourly basis to assist with the work cell simulation.

Publications

Two conference papers were presented and published in the respective proceedings:
References

W. Wälischmiller and H.-Y. Lee: Application of the Telbot Robot in Hazardous Environments
Nuclear Plant Journal Editorial Archive, September-October 1996.

W. Wälischmiller and H.-Y. Lee: Development and Application of the Telbot System – A New
Tele Robot System Proc. 11th CISM-IFToMM Symposium, Udine, Italy, 1996.

Shigeki Kudomi, Hironao Yamada, Takayoshi Muto: Development Of A Hydraulic Master-Slave

C.A. Balafoutis, R. Patel, P. Misra: Efficient Modeling and Computation of Manipulator Dynamics
1988, p. 665-676.

I-Ming Chen, Guilin Yang, In-Gyu Kang: Numerical Inverse Kinematics for Modular

Beis, J. S. and Lowe, D. G., “Shape Indexing Using Approximate Nearest-Neighbor Search in
High-Dimensional Spaces”, IEEE computer Vision and Pattern Recognition, pp. 1000 – 1006,
1997.

Chi, Zheru et al. ”Fuzzy Algorithms: with Applications to Image Processing and Pattern
OBJECT RECOGNITION OVER AN EXPANDED RANGE OF VIEWING ANGLES USING INDEXING METHODS

Jae-Kyu Lee, Ph.D. Student
Department of Mechanical Engineering
University of Nevada, Las Vegas
Las Vegas, NV 89154-4027, USA
E-mail: jklee@me.unlv.edu

Georg F. Mauer, Professor
Department of Mechanical Engineering
University of Nevada, Las Vegas
Las Vegas, NV 89154-4027, USA
E-mail: mauer@me.unlv.edu

Abstract

The main objective of the research presented here is the minimization of the data set required for the reliable recognition of arbitrarily positioned three-dimensional objects. The analytical framework for studying 3-D object acquisition and recognition from 2-D images is based on invariant indexing. The objective is to identify a 3-D object from 2-D images, taken from arbitrary spatial points of view. The geometric relation between 3D-object and the resulting 2D image is modeled as an affine transformation. Objects are modeled as sets of characteristic spatial points, such as corner and edge points, which are stored in a hash table. In a new view, known objects are quickly identified, and their orientation is estimated as long as the viewing angle deviates not too much from the angle at which the model was generated. Quantitative analysis about the range of permissible viewing angle variations and confidence intervals are presented.

Introduction

Indexing employs a priori stored information about the object models, in order to quickly eliminate non-compatible model-scene feature matches during recognition. Hence, only the most feasible matches are considered, that is, the matches where the model features could have projected to the scene features. Indexing-based methods usually employ a hash scheme to efficiently store and retrieve information about the models into a hash table. During preprocessing, groups of model features are considered, and a description for each is computed and stored in the indexed location. During recognition, groups of scene points are used to access the hash table. An invariant index computed from a group of model features remains unchanged regardless of changes in the appearance of the model regardless of viewpoint changes. Geometric hashing [1] uses affine invariants for the recognition of planar objects (2D). Here, non-collinear triplets of points establish bases from which the coordinates of other, coplanar points are found and entered into the table. At run-time, the corresponding model pattern is readily detected from any correct basis triple chosen from among the image points. While this technique differs substantially from other indexing methods in that voting overcomes some of the difficulty with bin boundaries, Grimson and Huttenlocher [1] [2] note that performance is poor even with small amounts of noise and clutter. This is due to the hash table becoming overloaded with entries, and an indication that the use of higher-dimensional spaces is important. Memory usage from the redundant storage of models is also excessive. The method of Clemens and Jacobs [3], while using hash tables, did not assume invariant feature sets. They derived 4D and 6D index spaces from general 3D point sets. However, the difficult task of feature grouping was done manually and the model database and index spaces were again too small for hashing inefficiency to be noticeable.

Building indexing schemes for general 3D objects using invariant indexing is not possible in general, since it has been shown that no general-case invariant exists for single views of general three-dimensional point sets. As a result, model based invariant indexing has been proposed. This invariance can be learned from several images of the object. The basic idea is that a function can be constructed for each group of model features that, given a group of image features, evaluates to true if and only if the model group could project to the image group. Below, we discuss several prominent indexing methods: Forsythe et al [6][7] outlined several types of projective invariant...
feature for indexing planar objects viewed in arbitrary 3D orientations. Their experiments were carried out using a 2D-index space generated by pairs of coplanar conic sections. Rothwell et al. [7] used 4D indices defined by area moments of planar curves that were first transformed into canonical reference frames. In each of these methods, the dimensionality of the spaces and the number of models were too small for the inefficiency of hashing to be critical. Stein and Medioni presented a method suitable for 2D from 2D [8] and 3D from 3D [9] indexing. The novel aspects of this work included the use of multiple levels of smoothing to improve segmentation. Beis and Lowe [10] provide evidence that, no matter how coarse the binding, hash table performance is poor in high-dimensional spaces. In [11], Califano and Mohan suggest the use of higher-dimensional spaces in indexing. Their analysis indicates a significant reduction in recognition time by increasing the size of the feature vectors.

**Approach** - Index-based recognition systems compute invariants from an image and then use them to index in a hash table. During model acquisition, the locations of the hash table entries can be used for pose recovery. During recognition, the models listed in the indexed entries are collected into a list of candidate models. The most likely candidate models are selected based on the number of times they were indexed. Recognition may include pose computation and verification. One well-known index-based approach is geometric hashing [1, 2], which handles 2-D shapes under similarity (affine) transforms. Three feature points in the image are chosen as a reference frame or basis, and the orthogonal (affine) coordinates \((\alpha, \beta)\) of every other model point in that basis are used for indexing. Each indexed entry is updated with the basis vectors and with a reference to the corresponding object model. This process is repeated using each triplet of model points as a basis. Object models are recognized directly from sampled image points without grouping or correlating any intermediate geometric features or subparts.

The recognition sequence is then:

1. **Feature Preprocessing:** From a 3D model or scene, extract a set of non-collinear triplets that can be represented as a set of points in the invariant space.
2. **Invariant Recovery:** If Euler rotation coefficients establishing the rotation, translation, and scaling information in invariant space are known, then align them together. If not, apply a probabilistic peaking method to predict their invariant.
3. **Evaluation:** Determine length ratios and angles for all non-collinear triplet point sets.
4. **Verification:** Verify the initial match by verifying geometric characteristics of other points with the model geometry.

**Orthographic Projection** - Given an object \(O \subset R^3\), we consider two sets of images of \(O\), the set of paraperspective images and the set of affine images of \(O\). The set of paraperspective images of \(O\), denoted by \(P\), contains the images of \(O\) obtained by applying a rigid transformation to \(O\) followed by a paraperspective projection with the reference point set in any position. Suppose \(O\) is transformed by a rotation \(R \in SO(3)\) and translation \(t \in R^3\) followed by a paraperspective projection \(\Pi_0\) with the reference point at \(P_0\). Then the projected coordinates of a point \(P_i \in O\) are \(q_i = \prod R(P_i + t)\), where \(t = [t_x, t_y, t_z]^T\) is the translation vector. We define \(r_{ij}\) \((1 \leq i, j \leq 3)\) as the components of the rotation matrix, \(R\), in Z-Y-Z Euler rotation. An affine image, \(A\), of \(O\) is obtained by applying a 3D affine transformation to \(O\) followed by an orthographic projection. We restrict \(A\) to include only non-degenerate affine transformations (that is, an affine transformation with a non-singular linear part). Suppose \(O\) is transformed by a \(3 \times 3\) non-singular linear matrix \(A \in GL(3)\) and translation \(t \in R^3\) followed by an orthographic projection \(\Pi_0\). Then, the projected coordinates of a point \(P_i \in O\) are given by \(q_i = \prod A(P_i + t)\).

The relation can be expressed as

\[
\begin{bmatrix}
X \\
Y \\
Z
\end{bmatrix} = \begin{bmatrix}
r_{11} & r_{12} & r_{13} \\
r_{21} & r_{22} & r_{23} \\
r_{31} & r_{32} & r_{33}
\end{bmatrix} \begin{bmatrix}
x \\
y \\
z
\end{bmatrix} + \begin{bmatrix}
p_1 \\
p_2 \\
p_3
\end{bmatrix}
\]

(1)
In orthographic projection (see Fig. 1), an object point coincides with its image point (i.e., \( x_i = X, y_i = Y \)). Three points suffice to determine the invariant between model image and scene image. The invariant between model image point \((x_i, y_i)\) and scene image point \((x'_i, y'_i)\) in Fig. 1 results as:

\[
\begin{pmatrix}
  x' \\
  y'
\end{pmatrix}
= \begin{pmatrix}
  r_{11} & r_{12} & x_i \\
  r_{21} & r_{22} & y_i
\end{pmatrix}
+ \begin{pmatrix}
  r_{31} \\
  r_{32}
\end{pmatrix}
\begin{pmatrix}
  z_i
\end{pmatrix}
\]

Premultiplication of both sides by \([r_{23}, -r_{13}]\) eliminates the \(z_i\) terms. We obtain:

\[
\begin{pmatrix}
  r_{23} & 0 & 0 & 0 \\
  0 & r_{32} & 0 & 0
\end{pmatrix}
\begin{pmatrix}
  x_i \\
  y_i
\end{pmatrix}
+ \begin{pmatrix}
  r_{31} \\
  r_{32}
\end{pmatrix}
\begin{pmatrix}
  z_i
\end{pmatrix}
= \begin{pmatrix}
  x'_i \\
  y'_i
\end{pmatrix}
\]

Equation (3) describes the invariant between model image point \(p(x_i, y_i)\) and scene image point \(p'(x'_i, y'_i)\) for orthographic projection, assuming the 3D rotation is known. Under orthographic or weak perspective projection without scaling \((s = 1)\) equation (3) becomes

\[
r_{23}x'_i - r_{13}y'_i + r_{32}x_i - r_{31}y_i = 0
\]

Equation (4) is linear and homogeneous in the four unknowns \(r_{13}, r_{23}, r_{31}, r_{32}\). \(N\) point correspondences result in \((N - 1)\) equations. Therefore, if \(N \geq 3\), and assuming that the points are not collinear, we can solve the set of equations (4) to obtain \(r_{13}, r_{23}, r_{31}, r_{32}\) within a scale factor.

From Equ.(4), the Euler angles \(\psi, \omega, \gamma\) are obtained as:

\[
\tan(\psi) = \frac{r_{23}}{r_{13}} \quad \tan(\omega) = \sqrt{1 + \left(\frac{r_{23}}{r_{13}}\right)^2} \quad \tan(\gamma) = -\frac{r_{32}}{r_{31} / r_{13}}
\]

Equation (4) is not linearly independent; so there are countless solutions for angle \(\omega\). Once we find the rotation \(R\), \(z_i\), and \(z'_i\) are found from Equation (2) as

\[
\begin{pmatrix}
  z_i
\end{pmatrix}
= \begin{pmatrix}
  r_{32}x_i + r_{31}y_i + r_{33} z_i
\end{pmatrix}
\]

Recovery of View Angle with Probabilistic Peaking - Rotation or depth information is usually not available. While there is no affine or projective invariant for general three-dimensional point sets, one can observe relative maxima for the probability densities over a significant range of angles and ratios of lengths in images at the values of features in the model. Thus a large range of viewing directions exists over which these feature values change only slightly. The probabilistic peaking method assumes that angles and ratios of distances between points in the model do not vary much when projected onto the image as the viewpoint varies over much of the viewing sphere. The resulting probability density functions have a strong peak as a Dirac delta (Unit Impulse) function at the preprojection (model) value so that we can build a probabilistic indexing system from point sets in 3D.

Recovery of Angle Ratio - Orthographic projection is appropriate when, as in most practical cases, the distance between camera and object is much larger than the object’s dimensions. The elevation angle, \(\sigma\), is the angle...
between frames $Z$ and $Z'$: The azimuth angle $\tau$ (see Fig. 2) is the rotation of the Z-axis. When $\sigma = 0$, the 3D object looks like a two dimensional quadrilateral shape independent of $\tau$. When viewed from $Z'$, the appearance of the 3D polyhedral object changes with respect to every angle. Figure 2 defines the geometrical terms for similarity analysis between a 2D reference image at camera position 1, and the image from position 2 at angles of $\sigma$ and $\tau$. The change of viewing angle results in changes of the included angles between base vector pairs $\lambda$ and $\delta$ in Fig. 3. The relationship between $\delta$, $\lambda$, and $\sigma$, $\tau$ is described by Equation (7). Figure 3 shows the relation for an angle $\delta = 45^\circ$.

We note that the deviation of $\lambda$ from its origin $\delta$ as a function of $\tau$ increases with $\sigma$. Also along the azimuth and elevation angles, $\sigma$ and $\tau$, deviate from the reference by less than 30°, the indexing vector scales almost the same as 2D since the curves remain near the $\sigma = 0$ line normal to the image plane, with deviations below 5%.

For ranges $\sigma (-30^\circ < \sigma < 30^\circ)$ and azimuth angle $\tau (-30^\circ < \tau < 30^\circ)$, the variations of the angle ($\lambda/\delta$) ratio were evaluated. The probability density has a maximum at $\log(\lambda/\delta) = 0$, indicating a high probability for close matching between the values $\delta$ and $\lambda$.

**Recovery of Distance Ratio** - An analogous estimation technique can be applied to the probability density of projected distances. We denote the origin pairs as the reference vector set $\{a_1, a_2\}$ and the projected vector set by $\{b_1, b_2\}$ as in Fig. 2. The length ratio $t$ is $t = \frac{\text{b_1}}{\text{a_1}}$.

Since the orthographic projection is linear, the lengths of $a_1$ and $a_2$ do not influence the ratio $t$. Using a minimum square error quality criterion, the probability density of $t$ can be approximated by the function $p_t (\log t)$

$$p_t(\log t) = x_1 \exp\left(-\frac{x_2}{x_1} |\log t|\right) + x_3 \exp\left(-\frac{x_4}{x_3} |\log t|\right)$$

(8)

We note that as long as $|\log(\lambda/\delta)| > 0.3$, the errors in estimating length ratios and angles remain below 20%. Figures 6(a) and (b) show the indexing value errors. Angle measurements are less sensitive to viewing angle variations, with $\alpha$ and $\beta$ below 5 percent in Figure 4 (c). However, in Figure 6 (b), length ratios $u$ and $v$ match poorly with up to 50 percent error. We therefore conclude that the viewing angle should not deviate by more than 30 degrees from the reference in either elevation or azimuth directions.
Fig. 3  Recognition results of angle and length in polyhedral synthetic object for $(\sigma, \tau > 30^\circ)$

Application to Real Images - A series of real images of a stapler at $30^\circ$ elevation angle and a reference image normal to the viewing plane were recorded. Each image has a commonality of top surface and small changes in the topside view. The interesting points were taken from the top surface of stapler image. Figure 4 shows recognition results between the surrounding test images and the center reference image. The angle and length ratios in Figure 4 (a) and (b), respectively show their magnitudes below 0.3. The confidence levels of indexing results are on average above 80%, see Figures 4 (c) and (d).

Conclusion
An indexing-based method for the detection of objects contained in camera images has been presented. 3D objects are characterized by a set of characteristic points such as corners and edges, and by the geometric relations between these points. The invariant indexing method permits for recovery of the spatial viewing angles and length ratios. Analytical and actual imaging examples were presented. Good agreement between model and scene object is ensured as long as the viewing angle deviation between both remains below $30^\circ$.

Acknowledgment
The authors gratefully acknowledge financial support for this work by the DOE Advanced Accelerator Applications (AAA) research program.

References and Bibliography


