PREMIO: An Overview

Octavia I. Camps, Linda G. Shapiro, and Robert M. Haralick

Intelligent Systems Laboratory
Electrical Engineering Department, FT-10
University of Washington
Seattle, WA 98195, U.S.A.

Abstract

A model-based vision system attempts to find a correspondence between features of an object model and features detected in an image. Most feature-based matching schemes assume that all the features that are potentially visible in a view of an object will appear with equal probability. The resultant matching algorithms have to allow for "errors" without really understanding what they mean. **PREMIO** is an object recognition/localization system under construction at the University of Washington that attempts to model some of the physical processes that can cause these "errors". PREMIO combines techniques of analytic graphics and computer vision to predict how features of the object will appear in images under various assumptions of lighting, viewpoint, sensor, and image processing operators. These analytic predictions are used in a probabilistic matching algorithm to guide the search and to greatly reduce the search space. In this paper, which is a discussion of work in progress, we describe the PREMIO System.

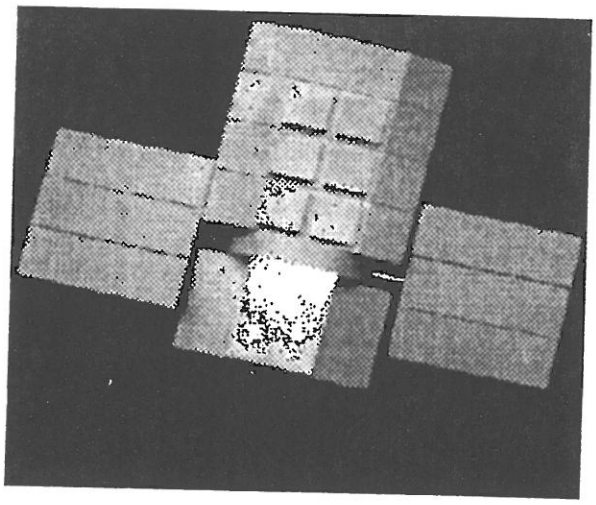
1 Introduction

The design of a *model-based vision system* able to recognize and locate an object in an image is an arduous process that involves trial and error experiments and requires a great deal of expertise from the designer. The automation of the design process is highly desirable; it would produce more effective procedures in less time, reducing the software cost of vision systems and expanding their use. Although previous work on automating the design of vision systems have had some success [6, 4, 16, 12, 17], there is still much work to be done on the object recognition and pose estimation problems. We believe that most of the limitations of the previous systems can be removed by the use of a more realistic model of the world. Hence, a better way of representing the interactions between the object representation schemes and the light sources and sensor properties must be found.

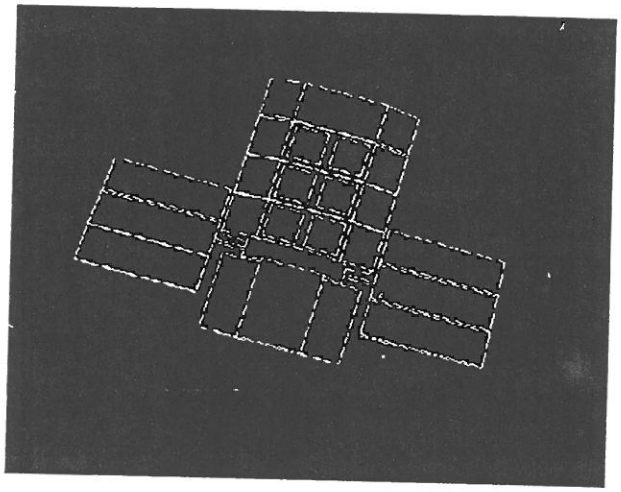
An example of the difficulties that a working vision system must address is illustrated in figure 1. Figure 1 (a) shows a grayscale image of a scaled-model of the satellite "Solarmax". Figure 1 (b) shows a naive prediction, which does not take into account the lighting and sensor characteristics of the edges that would be detected by an edge detector applied on the image of figure 1 (a). Figure 1 (c) shows the actual output of an edge detector where several of the predicted edges are fragmented or missing altogether. Knowledge of the degree to which each edge boundary might break up under different lighting and viewing conditions is essential. This knowledge ensures that the inductive matching phase does not have incorrect expectations that cause the search to look for something that does not exist and that the deductive hypothesis verification phase can employ a proper statistical test in which assumptions about what *should* be there match the reality of what *is* there.

2 PREMIO: A Model–Based Vision System

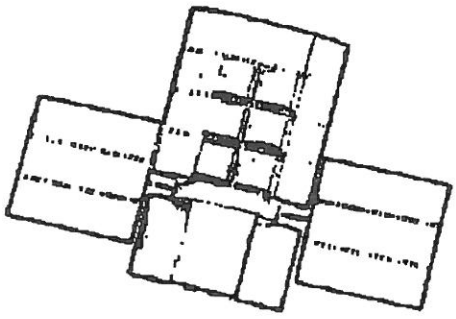
Most feature-based matching schemes assume that all the features that are potentially visible in a view of an object will appear with equal probability. The resultant matching algorithms have to allow for "errors" without really understanding what they mean. **PREMIO** (**PRE**diction in **M**atching **I**mages to **O**bjects) is an object recognition/localization system under construction at the University of Washington [8] that attempts to model some of the physical processes that can cause these "errors". PREMIO uses CAD models of 3D objects and knowledge of surface reflectance properties, light sources, sensors characteristics, and the performance of feature detectors to build a model called the *Vision Model*. The Vision Model is used to generate a model called the *Prediction Model* that is used to automatically generate vision algorithms. The system is illustrated in Figure 2. PREMIO's Vision Model is a more complete model of the world than the ones presented in the literature. It not only describes the object, light sources and camera geometries, but it also models their interactions. The Vision Model has five components: (1) a 3D topological model of the possible objects, describing their geometric properties and the topological relations between their faces, edges, and vertices; (2) a surface physical model, formed by a general model of the light reflection of surfaces and the physical characteristics describing their materials; (3) a light source and sensor geometrical model, representing their configuration in space; (4) a light source



(a) Solarmax grayscale image.



(b) Edge prediction without taking lighting and sensor into account.



(c) Output of an edge operator.

Figure 1: Problems in Feature Prediction.

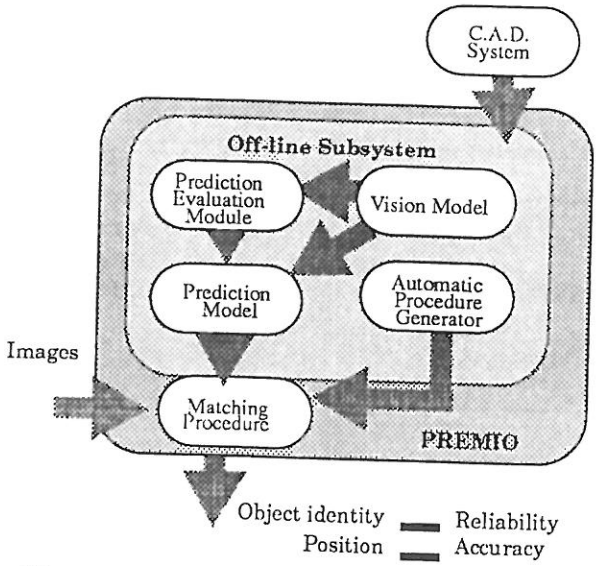


Figure 2: PREMIO: A Model-Based Vision System

and sensor physical model, describing their physical characteristics, and (5) a detector model describing the performance of the feature detectors available to the system.

The system has two major subsystems: an offline subsystem and a online subsystem. The offline subsystem, in turn, has three modules: a Vision Model generator, a feature predictor, and an automatic procedure generator. The Vision Model generator transforms the CAD models of the objects into their topological models and incorporates them into the Vision Model. The feature predictor uses the Vision Model to predict and evaluate the features that can be expected to be detected in an image of an object. The output of the Prediction Module is organized as the Prediction Model. The automatic procedure generator takes as its input the Prediction Model and generates the matching procedure to be used. The online subsystem consists of the matching procedure generated by the offline subsystem. It uses the Vision Model, the Feature Prediction Model, and the input images, first, to hypothesize the occurrence of an object and estimate the reliability of the hypotheses, and second, to determine the object position relative to the camera and estimate the accuracy of the calculated pose. We discuss each in turn.

3 The Vision Model

The Vision Model in a machine vision system is a *representation* of the world where the system works. A representation is a set of conventions about how to describe entities. Finding an appropriate representation is a major part of any problem-solving effort, and in particular in the design of a machine vision system.

The entities that must be described by our representation of the world are the objects to be imaged and

the characteristics that these images will have. These characteristics depend on: the geometry of the object; the physical characteristics of the object surfaces; the position of the object with respect to the sensors; the light sources and other objects; the characteristics of the light sources and the sensors, and ultimately, the characteristics of the device that "observes" the image.

3.1 Object Models

PREMIO assumes that it has available PADL2 CAD models of all the possible objects to be imaged. PADL2 is a constructive solid geometry (CSG) modeler designed by H. B. Voelcker and A. G. Requicha at the University of Rochester. Its primitives are spheres, cylinders, cones, rectangular parallelepipeds, wedges and tori.

PREMIO's object model is a hierarchical, relational model similar to the one proposed in [27]. The object model is called a *topological object model* because it not only represents the geometry of the objects but also the relations among their faces, edges, and vertices.

The model has six levels. A world level that is concerned with the arrangement of the different objects in the world. An object level that is concerned with the arrangement of the different faces, edges and vertices that form the objects. A face level that describes a face in terms of its surfaces and its boundaries. A surface level that specifies the elemental pieces that form those surfaces and the arcs that form the boundaries. Finally, a 1D piece level that specifies the elemental pieces that form the arcs.

The type of surfaces that a PADL2 model can have are the surfaces of its primitives: planes, spheres, cylinders, cones and tori. To represent these surfaces PREMIO uses the same representation that PADL2 uses: an implicit mathematical expression that represents the corresponding primitive in a "natural" coordinate system that makes this expression as simple as possible. An object modeled with PADL2 can only present boundary arcs that result from the intersection of its primitives. To represent these curves we also followed PADL2 choice: a parametric expression for each coordinate, with a range interval for each parameter, that represents the curve in a "natural" coordinate system that makes these expressions as simple as possible.

To create the topological object model from the PADL2 model, PREMIO uses the boundary file routines provided by PADL2. These routines give access to all the information concerning the face surfaces and the boundary arcs of the objects, but do not provide a direct way to extract the boundary, edge and vertex information that we need. To find the boundaries of a face, its arcs must be grouped together to form closed loops. This can be done using the algorithm developed by Welch [31] to find closed loops in an undirected graph. At the same time the edge and the vertex information can be updated. The edge relation provides a way to relate two faces that have an arc in common, while the vertex relation relates all the arcs that have a vertex in common. These two relations are very useful in the prediction of image features.

3.2 Object Surface Model

Given the physical properties of a material, it is possible to predict the properties of images of this material. Different materials reflect the light in different ways, producing different intensity values in the image. A reflection model of a surface is a series of equations designed to predict the intensity values of points in a scene. Given the light sources, the surface, and the position of the observer, the model describes the intensity and spectral composition of the reflected light reaching the observer. The intensity of the reflected light depends on the intensities, sizes, and positions of the light sources and on the reflecting ability and surface properties of the material. The spectral composition of the reflected light depends on the spectral composition of the light sources and on the wavelength-selective reflection of the surface.

The light reflected by a small region of a material can be broken down into three components: ambient, diffuse and specular. The ambient component describes the amount of light reaching the surface by reflection or scattering of the light sources or other background illuminators. Usually, ambient light can be assumed to be equal for all points on the surface and is reflected equally in all directions.

Most real surfaces are neither ideal specular (mirror-like) reflectors nor ideal diffuse (Lambertian) reflectors. Buchanan [7] has evaluated several reflectance models and concluded that Cook and Torrance's model [10] is the most accurate when the incident light is completely unpolarized. However, in general light is partially polarized. Yi [32] derived an extension of Cook's model for polarized light. PREMIO uses this model.

3.3 Light Sources and Sensors Models

Image formation occurs when a sensor registers radiation that has interacted with physical objects. Hence, it is important to include the light sources and sensor models in our vision model. A light source model must describe its position in space, its size and shape, and its wavelength components. A sensor model must describe its position in space, its response to the radiation input, and its resolution. In the offline system of PREMIO, the sensor and the light source positions are known. The sensor and light sources are placed on the surface of a sphere centered at the origin of the object coordinate system, called the reference sphere. The points on the reference sphere constitute a continuous viewing space. The viewing space is sampled [32] in a way such that the distance between any two neighboring points in the discrete viewing space is approximately the same.

The image intensity of a given point P in a given surface is given by [32]:

$$I = \int CSQ(\lambda) d\omega \vec{N} \cdot \vec{L} (R_{\parallel}(\lambda) J^i_{\parallel}(\lambda) + R_{\perp}(\lambda) J^i_{\perp}(\lambda)) d\lambda \quad (1)$$

where $\vec{N}$ is the unit normal to the given surface at P, $\vec{L}$ is the unit vector in the direction of the light source

from P, C is the lens collection factor, S is the sensor responsivity, Q is the spectral distribution of the illumination source, ω is solid angle, $J^i_{\parallel}$ and $J^i_{\perp}$ are the illumination intensities of the parallel and perpendicular polarized incident light, and $R_{\parallel}$ and $R_{\perp}$ are the bi-directional functions for the parallel and perpendicular polarized incident light.

The lens collection factor, C, is given by [32]:

$$C = \frac{\pi}{4}\left(\frac{a}{f}\right)^2 \cos^4 \alpha \ . \qquad (2)$$

where f is the focal distance of the lens, a is the diameter of the lens, and α is the angle between the ray from the object patch to the center of the lens. The sensor responsivity S, is in general a function of the wavelength of the incident light. However, for monochromatic sensors it can be approximated to one, regardless of the wavelength of the incident light.

4 Feature Prediction Module

Given a vision model representing the world, the goal of the prediction module is threefold: (1) it has to predict the features that will appear on an image taken from the object from a given viewpoint and under given lighting conditions; (2) it has to evaluate the usefulness of the predicted features, and (3) it has to organize the data produced by (1) and (2) in a efficient and convenient way for later use. Our approach to this is analytic.

4.1 Predicting Features

There are two different approaches to the use of CAD-Vision models for feature prediction: synthetic-image-based prediction and model-based feature prediction.

Synthetic-image-based feature prediction consists of generating synthetic images and extracting their features by applying the same process that will be applied to the real images. Amanatides [1] recently surveyed different techniques used in realistic image generation. A particularly powerful technique used to achieve realism is ray casting: cast a ray from the center of projection through each picture element and identify the visible surface as the surface that intersects the ray closer to the center of projection. Bhanu et al [3] used ray casting to simulate range images for their vision model.

Model-based feature prediction uses models of the object, of the light sources and of the reflectance properties of the materials together with the laws of physics to analytically predict those features that will appear in the image for a given view without actually generating the gray-tone images. Instead, only data structures are generated. This is a more difficult approach, but it provides a more computationally efficient framework suitable for deductive and inductive reasoning. This is the approach used by PREMIO.

4.2 Model-Based Feature Prediction

The model-based feature prediction task can be divided into three steps: The first step is to find the edges that would appear in the image, taking into account only the object geometry and the viewing specifications. The result is similar to a wireframe rendering of the object, with the hidden lines and surfaces removed. The second step is to use the material reflectance properties and the lighting knowledge to find the contrast values along the edges in a perspective projective image, and to predict any edge that may appear due to highlighted or shaded regions on the image. The third and last step is to interpret and group the predicted edges into more complex features such as triplets, corners, forks, holes, etc.

4.2.1 Wireframe Prediction

The problem of determining which parts of an object should appear and which parts should be omitted is a well-known problem in computer graphics. A complete survey of algorithms to solve the "Hidden-Line, Hidden-Surface" problem can be found in [29]. A particularly efficient way of solving this problem is using an analytical approach, by projecting the object surface and boundary equations onto the image plane and determining whether the resulting edges are visible or not. This approach obtains the edges as a whole, as opposed to the ray casting approach, which finds the edges pixel by pixel. The aim of the solution is to compute "exactly" what the image should be; it will be correct even if enlarged many times, while ray casting solutions are calculated for a given resolution. Hence this is the preferred method for our application.

In order to analytically predict a wireframe we need to introduce the following definitions:

Def. 4.1 A *boundary* is a closed curve formed by points on the object where the surface normal is discontinuous.

Def. 4.2 A *limb* is a curve formed by points on the surface of the object where the line of sight is tangent to the surface, i.e. perpendicular to the surface normal.

Def. 4.3 A *contour* is the projection of a limb or a boundary onto the image plane.

Def. 4.4 A *T-junction* is a point where two contours intersect.

Def. 4.5 A *cusp point* is a limb point where the line of sight is aligned with the limb tangent.

The edges in an image are a subset of the set of contours. A piece of a contour will not appear in the image if its corresponding boundary or limb is part of a surface that is partially or totally occluded by another surface closer to the point of view. Since the visibility of a contour only changes at a cusp point or a T-junction point, it follows that to find the edges on the image the following steps have to be taken: (1) find all

the limbs and cusp points, (2) project the boundaries and limbs to find the contours and all the T-junctions and (3) determine the visibility of the contours by finding the object surface closest to the point of view at each T-junction and cusp point.

Finding Limbs and Cusp Points

To find the analytical expressions for the limbs and cusp points, PREMIO uses an approach similar to the one used in [22], but designed for PADL2-modelable objects instead of generalized cylinders.

Let P_0 with object coordinates (X_0, Y_0, Z_0) be the projection center and let P with object coordinates (X, Y, Z) be a point on a limb on the surface S defined by the implicit equation $f(X, Y, Z) = 0$. Then, the vector of sight $\vec{v}$ from P_0 to P is given by:

$$\vec{v} = (X - X_0, Y - Y_0, Z - Z_0) \tag{3}$$

and the normal $\vec{N}$ to the surface S is given by:

$$\vec{N} = \left(\frac{\partial f}{\partial X}, \frac{\partial f}{\partial Y}, \frac{\partial f}{\partial Z}\right) \tag{4}$$

In order for P to belong to the limb curve, P must be on the surface S and the line of sight must be perpendicular to the normal $\vec{N}$ at P. Hence the limb equations are given by:

$$\begin{cases} \vec{v} \cdot \vec{N} &= 0 \\ f(X,Y,Z) &= 0 \end{cases} \tag{5}$$

Once the limb equations are solved, a limb can be expressed in a parametrized form:

$$\begin{cases} X &= X(t) \\ Y &= Y(t) \\ Z &= Z(t) \end{cases} \quad t_{min} \le t \le t_{max} \tag{6}$$

Then, the tangent vector $\vec{T}$ to the limb is given by:

$$\vec{T} = \left(\frac{\partial X}{\partial t}, \frac{\partial Y}{\partial t}, \frac{\partial Z}{\partial t}\right) \tag{7}$$

Since a cusp point C is a limb point where the line of sight is aligned with the limb tangent, its coordinates must satisfy the following equations:

$$\begin{cases} \vec{T} \times \vec{v} &= 0 \\ X &= X(t) \\ Y &= Y(t) \\ Z &= Z(t) \end{cases} \quad t_{min} \le t \le t_{max} \tag{8}$$

This procedure is performed in $O(s)$ time where s is the number of curved surfaces of the object.

Finding the contours and T-junctions

To find the contours, the limbs and boundaries of the object are projected onto the image plane; to find the T-junctions the intersections of the contours are found. The intersection detection problem for n planar objects has been extensively studied and it can be solved in $O(n \log n + s)$ time [23], where s is the number of intersections. In our case, the objects are the set of contours. For PADL2 primitives, the limb curves are either circles or straight lines, while the boundaries can be either straight lines, conics or more complex curves. Since the perspective projection of a straight line is another straight line, and the perspective projection of a conic is another conic, we can find a closed form solution for the intersections between the contours that result from projecting straight lines and conics. To find other type of T-junctions, a numerical approach must be used.

Determining Visibility

The next step is to determine the edges and surfaces that are hidden by occlusion. Appel [2], Loutrel [19], and Galimberti and Montanari [11] have presented similar algorithms for analytical hidden line removal for line drawings. They define the *quantitative invisibility* of a point as the number of relevant faces that lie between the point and the camera. Then, the problem of hidden line removal reduces to computing the quantitative invisibility of every point on each relevant edge. The computational effort involved in this task is dramatically reduced by the fact that an object's visibility in the image can change only at a T-junction or at a cusp point. At such points, the quantitative invisibility increases or decreases by 1. This change can be determined by casting a ray through the point and ordering the corresponding object surfaces in a "toothpick" manner along the ray. Hence, if the invisibility of an initial vertex is known, the visibility of each segment can be calculated by summing the quantitative invisibility changes.

The quantitative invisibility of the initial vertex is determined by doing an exhaustive search of all relevant object faces in order to count how many faces hide the vertex. An object face is considered relevant if it "faces" the camera, i.e. its outside surface normal points towards the camera. A face hides a vertex if the line of sight to the vertex intersects the face surface and if the intersection point is inside the boundary of the face. To propagate the quantitative invisibility from one edge to another edge starting at its ending vertex, a correction must be applied to the quantitative invisibility of the starting point of the new edge. The complication arises from the fact that faces that intersect at the considered vertex may hide edges emanating from the vertex. This correction factor involves only those faces that intersect at the vertex. For an object with e edges, f faces, and with an average of 3 faces meeting at each vertex, the computational time needed to remove its hidden lines using this algorithm is $O(f + 2 \times 3 \times e)$.

4.2.2 Using Material and Lighting Knowledge

Boundaries of objects show up as intensity discontinuities in an image. A line segment that is potentially visible in a set of views of an object may appear as a whole, disappear entirely, or break up into small segments under various lighting assumptions depending upon the contrast along the edges and the detector

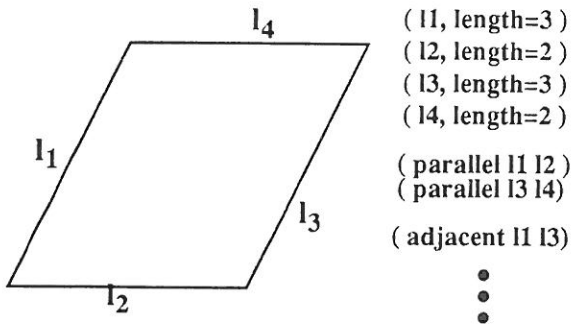


Figure 3: Feature and relationships example

characteristics. Hence, to complete the prediction process PREMIO needs to calculate the intensity values along the predicted wireframe.

The contrast at an edge point is computed as the difference in the intensity of the reflected light from two small neighboring patches at each side of the edge. These intensities, in turn, are obtained by using ray casting and the surface reflection model at a finite number of points along the edges. To represent the contrast along the edge, a contrast graph is fitted with piece-wise continuous polynomials using a regression analysis technique [32].

4.2.3 Interpreting and Grouping Features

A *feature* is an entity that describes a part of an object or an image. Simple features such as edges can be interpreted by themselves, or can be grouped to be considered as higher-level features. Matching *perceptual groupings* of features was suggested first by Lowe [20]. Henikoff and Shapiro [15] have found useful for object matching arrangements of triplets of line segments called *interesting patterns*. Other useful high-level features are junctions, and closed loops [21].

In PREMIO a feature is an abstract concept; it can be a point, an edge, a triplet, a hole, a junction, or a higher-level combination of any of these. A feature has a *type* that identifies it, a vector of *attributes* that represent its global properties, and a real number between 0 and 1 called its *strength*. The strength is a measurement of the confidence of the feature being of a particular type.

A feature participates in spatial relationships with other features. Each such relationship is represented by a *relational tuple*, which consists of a *type* specifying the relationship and a vector of *related features* that participate in that relationship. Associated with every relational tuple of features there is a real number between 0 and 1 called the *strength* of the relational tuple. The strength is a measurement of the confidence of the feature vector satisfying the specified relationship.

As a simple example, consider the parallelogram shown in figure 3. It can be described in terms of its four sides and the relationships among them. In this case, the features are the four sides of the parallelogram l_1, l_2, l_3 and l_4. Each side has associated the attribute *length* and it is related to the other three features by the relationships *adjacent* and *parallel*.

4.3 Evaluating Predicted Features

After a feature is predicted its potential utility must be evaluated. PREMIO uses the concepts of detectability, reliability and accuracy of a feature. The *detectability* of a feature is defined as the probability of finding the feature using a given detector on an image taken with a given sensor. Therefore, its value depends not only on the feature, but also on the sensor and detector models. The *reliability* of a feature is the probability of correctly matching the detected feature to the corresponding one in the model. Two features that look very similar to each other, should not be considered as very reliable since each of them can be mistakenly identified as the other. In general, the reliability is closely related to the distinguishability power of the feature; i.e. a unique feature immediately matches the model, and therefore is highly reliable. The feature *accuracy* is a measure of the error or uncertainty propagated from the detected feature to a geometric property of the object. This means that if, for example, we detect a straight line in the image, we want to bound the error of its location and orientation. Hence, the accuracy is calculated taking into account the sensor and detector models.

4.4 Output of the Predictor Module

For a given object and a given configuration of light sources, and sensors, the output of the predictor module is a hierarchical relational data structure similar to the one defined in section 3. This structure is called a *prediction* of the object. Each prediction contains a set of features, their attribute values such as detectability, reliability, and accuracy, and their originating three-dimensional features. The prediction has at least five levels: the image level, an object level, one or more feature levels, an arc level and a 1D piece level. The image level at the top of the hierarchy is concerned with the imaging conditions that generated the prediction, the general object position, and the background information. The object level is concerned with the different features that will appear on the image and their interrelationships. The feature levels describe the features in terms of simpler features, down to the arc level. The arc level describes the arcs in terms of 1D pieces. Finally, the 1D-piece level specifies the elemental pieces that form the arcs.

5 Using Prediction in Matching

The predictions that PREMIO produces are powerful new tools in recognizing and determining the pose of a 3D object. In order to take advantage of these tools, we have developed an entirely new matching algorithm, a

branch-and-bound search that explicitly takes advantage of the probabilities obtained during the prediction stage to guide the search and prune the tree. The matching algorithm represents a large theoretical effort that is actually independent of the PREMIO system, and it is fully described in [9]. The algorithm has been implemented as a C program and tested independently on data specifically generated to fit the abstract paradigm for the probabilistic search.

The matching algorithm can be thought of in two ways, as a relational matching algorithm and as a constrained branch-and-bound search. The theory behind branch-and-bound search is well known [18]. Relational matching has been expressed in several different formalisms. Early papers concentrated on graph or subgraph isomorphisms [30]. This led to many algorithms for discrete relaxation and the introduction of probabilistic relaxation [24]. The exact matching problem was generalized to the consistent labeling problem [14] and to the inexact matching problem [26]. This was extended further to the problem of determining the relational distance between two structural descriptions [28, 25]. Some recent related work includes structural stereopsis using information theory [5] . The present algorithm differs from all of these in its attempt to provide a solid theoretical probabilistic framework for the matching problem and the search.

5.1 Definitions and Notation

Models and images are represented by their features, the relationships among them, and the measurements associated with them. As in the consistent labeling formalism [14], we will call the image features *units* and the model features *labels*. The matching algorithm must determine the correspondences between the units and the labels. Formally, a *model* M is a quadruple $M = (L, R, f_L, g_R)$ where L is the set of model features or labels, R is a set of relational tuples of labels, f_L is the attribute-value mapping that associates a value with each attribute of a label of L, and g_R is the strength mapping that associates a strength with each relational tuple of R. Similarly, an *image* I is a quadruple $I = (U, S, f_U, g_S)$ where U is the set of image features or units, S is a set of relational tuples of units, f_U is the attribute-value mapping associated with U, and g_S is the strength mapping associated with S.

The relational matching problem is a special case of the pattern complex recognition problem [13]. An image is an observation of a particular model. Let $M = (L, R, f_L, g_R)$ be the model, and $I = (U, S, f_U, g_S)$ be the observed image. Not all the labels in L participate in the observation, only a subset of labels $H \subseteq L$ is actually observed. Furthermore, only the relational tuples of labels representing relationships among labels in H can be observed, and only a subset of them are actually observed. The set U consists of the unrecognized units. Some of the units observed in U come from labels in H; others are unrelated and can be thought of as clutter objects.

The relational matching problem is to find an unknown one-to-one correspondence $h: L \rightarrow U$ between a subset of L, H, and a subset of U, associating some labels of L with some units of U. The mapping h is called the *observation mapping*, and it must satisfy that the number of labels associated with units and the number of relations preserved in the observation are maximized. Notice that the matching process consists not only of finding the model M, but also of finding the correspondence h and its domain H, which are the explanation of why the model M is the most likely model. In general we seek to maximize the *a posteriori* probability $P\left(M, h \middle| I\right)$. That is, we want to maximize the probability of the model being M and the observation mapping being h, given that the image I is observed.

The relational matching problem requires a search procedure that can identify the model M and the mapping h such that $P\left(M, h \middle| I\right)$ is maximized. If the *relational matching cost* of an observation mapping h, $C(M, h, I)$ is defined by,

$$C\left(M, h, I\right) = -\log P\left(M, h, I\right) , \qquad (9)$$

then maximizing $P\left(M, h, I\right)$ is equivalent to minimize the relational cost of h.

The relational cost C can be broken down into five terms, each one representing a different aspect of the cost of the mapping [9]:

$$C\left(M, h, I\right) = C_M + C_U + C_S + C_{f_U} + C_{g_S} , \qquad (10)$$

where

$$\begin{aligned} C_M &= -\log P\left(M\right) , \\ C_U &= -\log P\left(U, h \middle| M\right) , \\ C_{f_U} &= -\log P\left(f_U \middle| U, M, h\right) , \\ C_S &= -\log P\left(S \middle| U, M, h\right) , \\ C_{g_S} &= -\log P\left(g_S \middle| U, f_U, M, h\right) . \end{aligned}$$

The cost C_M is the *model cost*. This is the cost associated with the model being considered, and it penalizes the selection of models whose prior probability of occurring, $P(M)$, is low.

The costs C_U and C_{f_U} are the *label–unit assignment costs*, and they evaluate how well the labels and units match through the mapping h. C_U is the part of the cost that penalizes for the differences of sizes between the set of observed features U and the set of features of the model L. C_{f_U} is the part of the cost that penalizes the "differences" between labels and their correspondent units. The costs C_U and C_{f_U} are given by [9]:

$$\begin{aligned} C_U &= -\log k_f - N_f \log k_f q_f , \\ C_{f_U} &= -\log\left(P\left(\rho(f_U \circ h, f_{L|_H}) \middle| M\right)\right) , \end{aligned} \qquad (11)$$

where $N_f = \#L + \#U - 2\#H$, $k_f > 0$ and $0 < q_f < 1$ are constants and are determined for each model from the predictions using regression analysis techniques, ρ is a suitable metric function, $f_U \circ h$ is the composition of f_U with h, and $f_{L|_H}$ represents the attribute-value mapping f_L restricted to the labels in the domain H.

The costs C_S and C_{g_S} are the *relational structural costs* and they evaluate how well the relationships among the labels are preserved by the mapping h. C_S is the part of the cost that accounts for the differences between the set of observed relationships S and the set of relationships of the model R. C_{g_S} is the part of the cost that penalizes the "differences" between the relational tuples of labels and their correspondent relational tuples of units. The costs C_S and C_{g_S} are given by [9]:

$$\begin{aligned} C_S &= -\log k_r - N_r \log q_r \ , \\ C_{g_S} &= -\log\left(P\Big(\rho(h \circ g_S, g_R)\Big|M\Big)\right) \ , \end{aligned} \quad (12)$$

where $N_r = \#(R - S \circ h^{-1}) + \#(S - R \circ h)$, $k_r > 0$ and $0 < q_r < 1$ are constants and are determined for each model from the predictions using regression analysis, $S \circ h^{-1}$ is the composition of S with the inverse mapping of h, h^{-1}, $R \circ h$ is the composition of R with h, ρ is a suitable metric function, and $g_S \circ h$ is the composition of g_S with h.

5.2 Partial Matching

Finding the full mapping h would require a full tree search. But, only a few correspondences between units and labels are needed to hypothesize a match between an object and a model and to estimate the object's pose. The number of correspondences needed is determined by the number of degrees of freedom that the matched features fix. Instead of finding the entire mapping h, we would like to find a partial match m that is a *restriction* of h, in the following sense:

Def. 5.1 Given two one-to-one mappings h and m, such that $\mathrm{Dom}(m) \subseteq \mathrm{Dom}(h)$, and $m(l) = h(l)$ for all $l \in \mathrm{Dom}(m)$, we say that the function h is an *extension* of the function m, and that the function m is a *restriction* of the function h. The *order* of the extension h with respect to m is the difference between the cardinalities of the sets $\mathrm{Dom}(h)$ and $\mathrm{Dom}(m)$.

Let $m: L \rightarrow U$ be a partial mapping assigning some labels to some units. The mapping m partitions the sets of features L and U into the set of *used* features in the match and the set of *residual* features – i.e., those not used in the match. Figure 4 gives a diagram of the sets L and U showing the partitions induced by a partial match m.

Let L^u be the set of used labels, L^r the set of residual labels, U^u the set of used units, and U^r the set of residual units induced by the partial mapping m. Consider the set $E_j = \{ext_j(m)\}$, of all the possible extensions of m of order j that assign some labels to some units. The maximum possible order of an extension of m is given by: $J = \min\{\#L^r, \#U^r\}$. The set

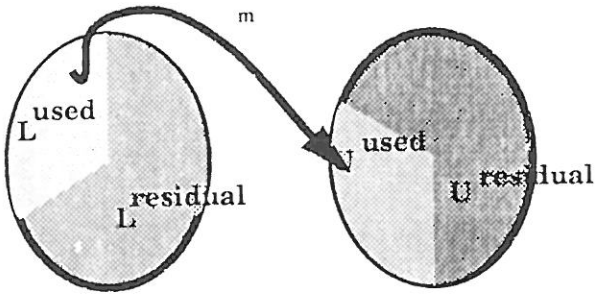


Figure 4: Partition of the sets of features induced by a partial match.

$E = \{ext(m)\}$ of all possible extensions of m can be expressed as the union of all the extensions of different orders: $E = \bigcup_{0 \le j \le J} E_j$, and its cardinal is given by:

$$\#E = \sum_{j=0}^{j=J} \left(\begin{array}{c} \#L^r \\ j \end{array} \right) \cdot \left(\begin{array}{c} \#U^r \\ j \end{array} \right) \cdot j! \ .$$

The probability that the "true" observation mapping h is an extension of a partial mapping m – that is the probability that the observation mapping h that maximizes the probability $P\Big(M, h, I\Big)$ belongs to the set $E = \{ext(m)\}$ of all possible extensions of the partial mapping m is given by [9]:

$$P\Big(M, (m, L^u), h \in E, I\Big) = \frac{P_M \cdot \mathrm{P}_S \cdot \mathrm{P}_{f_U} \cdot \mathrm{P}_{g_S}}{\mathrm{P}_U{}^2} \quad (13)$$

where,

$$\begin{aligned} P_M &= P\Big(M\Big) \\ \mathrm{P}_U &= k_f \sum_{j=0}^{j=J} \#E_j \, q_f{}^{N_{fj}} \\ \mathrm{P}_S &= k_f k_r \sum_{j=0}^{j=J} q_f{}^{N_{fj}} \sum_{h_i \in E_j} q_r{}^{N_{ri}} \\ \mathrm{P}_{f_U} &= k_f \sum_{j=0}^{j=J} q_f{}^{N_{fj}} \sum_{h_i \in E_j} P\Big(\rho(f_U \circ h_i, f_{L|_{U^u}}) \Big| M\Big) \\ \mathrm{P}_{g_S} &= k_f \sum_{j=0}^{j=J} q_f{}^{N_{fj}} \sum_{h_i \in E_j} P\Big(\rho(h_i \circ g_S, g_R) \Big| M\Big) \\ N_{fj} &= \#L^r + \#U^r - 2j \\ N_{ri} &= \#(R - S \circ h_i^{-1}) + \#(S - R \circ h_i) \ . \end{aligned}$$

Although the terms P_S, P_{f_U}, and P_{g_S} cannot be calculated unless all the possible extensions $h_i \in E$

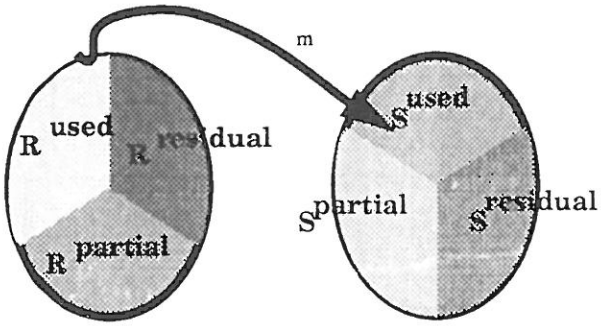
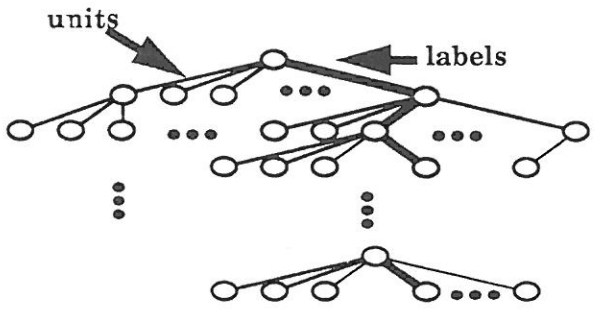


Figure 5: Partition of the sets of relational tuples induced by a partial match.



Search Space: All possible interpretations
Search State: A path in the tree

Figure 6: Search tree $\mathcal{T}$.

are considered, they can be *upper bounded* by values depending only on m and not on its extensions [9]. These upper bounds can be found by noticing that the partial mapping m induces a partition of the sets of relational tuples S and R into three types of sets: the set of *used* relational tuples, the set of *partially used* relational tuples, and the sets of *residual* relational tuples, depending on whether all, some, or none of the features in the feature vector of the tuple have been associated a correspondent through the mapping m. Figure 5 gives a diagram of the sets R and S showing the partitions induced by a partial match m. Then, it can be shown [9] that

$$P\Big(M, m, h \in E, I\Big) \leq P_{max} \tag{14}$$

where

$$P_{max} = P_M \cdot k_f k_r \sum_{j=0}^{j=J} \#E_j q_f^{N_{fj}} q_r^{N_{rmaxj}} \ . \tag{15}$$

where $N_{rmaxj} = \#R + \#S - R_{maxj} - S_{maxj}$, and R_{maxj} and S_{maxj} are the total number of relational tuples of labels and units with at most j labels or units without a correspondent in the mapping m.

5.2.1 Matching by Tree Search

The matching process can be thought of as a state space search through the space of all possible interpretations Σ. The state space Σ is called the *matching space* and it is defined as follows:

Def. 5.2 The *matching space*, Σ, is the state space of all possible interpretations, in which each state σ is defined by an observation mapping h_σ with degree of match $k_\sigma = \#\mathrm{Dom}(h_\sigma)$.

The search through the state space Σ can be achieved by doing an ordered search on a tree $\mathcal{T}$ such as the one shown in figure 6. Each node in $\mathcal{T}$ represents a unit and each of its branches represents an assignment of the unit to a label. A search state σ in Σ is represented by a path $\mathcal{P}$ in the tree $\mathcal{T}$. In the rest of the paper, the terms "path" and "partial mapping" will be used interchangeably.

A path $\mathcal{P}$ defines an observation mapping $m_{\mathcal{P}}$, and it has an associated cost $C_{\mathcal{P}} = C(m_{\mathcal{P}}, M, I)$ defined in equation (9). The matching process consists of finding the path $\mathcal{P}^*$ such that its associated observation mapping $m_{\mathcal{P}^*}$ has the least cost.

A match can be found by using the well known *branch-and-bound* tree search technique. In the standard branch and bound approach during search there are many incomplete paths contending for further consideration. The one with the least cost is extended one level, creating as many new incomplete paths as there are branches. This procedure is repeated until the tree is exhausted.

5.2.2 Improved Branch-and-Bound Search

Branch and bound search can be improved greatly if the path to be extended is selected such that a *lower bound* estimate of its the total cost is minimal. Those branches that have an estimated total cost greater than the maximum cost allowed can be pruned.

Let m be a partial mapping and m_1 be an extension of m. The relational matching cost of m_1 is given by $C_{m_1} = -\log P\Big(M, m_1, I\Big)$. An underestimate of C_{m_1} is found by finding an upper bound of $P\Big(M, m_1, I\Big)$. Let h^* be the true observation mapping. Since $h^* \in E$ is one of a set of disjoint events, the probability $P\Big(M, m, h^* \in E, I\Big)$ can be expressed as the sum of the probabilities of these events:

$$P\Big(M, m, h^* \in E, I\Big) = \sum_{h_i \in E} P\Big(M, h_i, I\Big) \ .$$

Hence, for an extension m_1 we have

$$P\Big(M, m_1, I\Big) \leq P\Big(M, m, h^* \in E, I\Big) \leq P_{max} \ . \tag{16}$$

Step 1: Initialization.
Form a queue $Q_{\mathcal{P}}$ of partial matches, and let $\mathcal{P}_0$ be the initial partial match.
Step 2: Iterate over current paths.
Until $Q_{\mathcal{P}}$ is empty, do
Begin
  $\mathcal{P}$:= FRONT($Q_{\mathcal{P}}$)
  m := partial mapping associated with $\mathcal{P}$
  C_m := relational cost of m
  Step 2.1: Test if $\mathcal{P}$ can be extended.
  If the path $\mathcal{P}$ can be extended,
    Begin
      Step 2.1.1: Select next label.
      Look for two tuples, one from R and one from S whose components are not all matched, that are compatible. Two relational tuples are compatible if they have the same number of features and they agree on the features that have been already matched. The relational tuples that are partially matched should be checked before than those that are not.
      Step 2.1.2 Extend the path
      For each $u \in U^r$, do
      Begin
        h_1 := path m extended with the pair (l, u).
        $\mathcal{P}'$:= path associated with the mapping h_1.
        C_{h_1} := relational cost of h_1.
        Γ_{h_1} := underestimate of the cost of the extensions of h_1.
        Step 2.1.2.1 Compare with ϵ.
        If $\Gamma_{h_1} \leq \epsilon$
        Begin
          Step 2.1.2.1.1 Finished?
          If FP($\mathcal{P}'$) = 6 and $C_{h_1} \leq \epsilon$
          Begin
            $\mathcal{P}'$ is a satisfactory match.
            Exit .
          End if.
          Step 2.1.2.1.2 Add $\mathcal{P}'$.
          BACK($Q_{\mathcal{P}}$) := $\mathcal{P}'$.
        End if.
      End for.
      Step 2.1.3 Resort the queue.
      Sort $Q_{\mathcal{P}}$ by underestimated cost.
    End if.
  End until.
Step 3: End of Algorithm
Announce failure.

Figure 7: Matching Algorithm

The matching algorithm is given in figure 7. The algorithm is being independently tested using controlled experiments designed under a rigorous experimental protocol [9]. So far, it has been tested on more than 4000 runs for models with five and seven labels, and with ordered binary and ternary relational tuples. Figure 8 is a plot of the ratio of the number of paths pruned to the total number of paths opened during the search. The graph shows that the use of the underestimate bound of the cost results in a high pruning ratio (from 30% to nearly 80% of the tree), and hence greatly reduces the computational time.

6 Summary and Future Work

Our research consists of two parallel activities: theoretical developments, and test of the resulting theory by

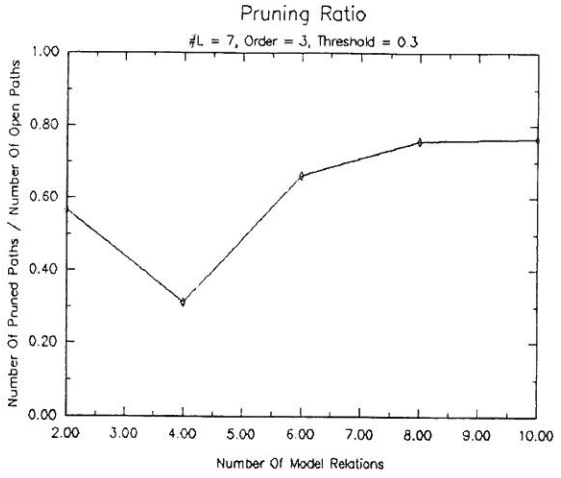


Figure 8: Pruning Ratio

the implementation of the **PREMIO** system, which is implemented as a set of routines with over 39000 lines of C language on a SUN system running Unix. **PREMIO** has two major subsystems: an offline subsystem and a online subsystem. The offline subsystem, in turn, has three modules: a vision model generator, a feature predictor, and an automatic procedure generator. We have proposed the use of a complete model of the world, the Vision Model, that incorporates PADL2 CAD models, surface reflectance properties, light sources, sensors, and processing models to symbolically predict the features that will appear on the images of the objects being modeled. The predictions are organized in a Prediction Model that produces the knowledge base of the probabilistic matching algorithm.

For the vision model, we have implemented a hierarchical, topological representation of the objects in the world, using as input their PADL2 CAD models. The prediction module can now predict line segments feature for objects with planar surfaces. The probabilistic matching algorithm is being tested independently using artificial data generated under a rigorous experimental protocol. The results obtained so far are promising in that a large percentage of the tree being searched is pruned by the matching procedure proposed. The remaining work is to integrate the parts of the system and test it on real image data. On the basis of our results so far, we expect PREMIO, when fully integrated, to solve many of the difficulties that most CAD–based vision systems encounter.

References

[1] J. Amanatides. Realism in computer graphics: A survey. *IEEE Computer Graphics and Applications*, 7:44–56, January 1987.

[2] A. Appel. The notion of quantitative invisibility. In *Proc. ACM National Conference*, pages 387–393, 1967.

[3] B. Bhanu, T.Henderson, and S.Thomas. 3-D model building using CAGD techniques. In *Proc. IEEE Computer Vision and Pattern Recognition*, pages 234–239, June 1985.

[4] R.C. Bolles and R.A. Cain. Recognizing and locating partially visible objects: The local-feature focus method. *Int. J. Robot. Res.*, 1(3):57–82, Fall 1982.

[5] K. L. Boyer and A. C. Kak. Structural stereopsis for 3-d vision. *IEEE Transactions on Systems, Man and Cybernetics*, PAMI–10(2):144–166, March 1988.

[6] R.A. Brooks. Symbolic reasoning among 3-D models and 2-D images. *Artificial Intelligence*, 17(1-3):285–348, 1981.

[7] C.G. Buchanan. Determining surface orientation from specular highlights. Master's thesis, Dep. Comp. Sc., Univ. of Toronto, Toronto, Ontario, Canada, 1986.

[8] Octavia I. Camps, Linda G. Shapiro, and Robert M. Haralick. PREMIO: The Use of Prediction in a CAD-Model-Based Vision System. Technical Report EE-ISL-89-01, Department of Electrical Engineering, University of Washington, 1989.

[9] Octavia I. Camps, Linda G. Shapiro, and Robert M. Haralick. A probabilistic matching algorithm for object recognition. Technical Report EE-ISL-90-08, Department of Electrical Engineering, University of Washington, 1990.

[10] R.L. Cook and K.E. Torrance. A reflectance model for computer graphics. *ACM Trans. on Graphics*, 1(1):7–24, January 1982.

[11] R. Galimberti and U. Montanari. An algorithm for hidden-line elimination. *Comm. ACM*, 12(4):206–211, April 1969.

[12] C. Goad. Special purpose automatic programming for 3D model-based vision. In *Proc. of the Image Understanding Workshop*, pages 94–104, June 1983.

[13] R.M. Haralick. The pattern complex. In Roger Mohr, Theo Pavlidis, and Alberto Sanfeliu, editors, *Structural Pattern Analysis*, pages 57–66. World Scientific Public. Co, 1989.

[14] R.M. Haralick and L. G. Shapiro. The consistent labeling problem: part i. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, PAMI–1(2):173–184, April 1979.

[15] J. Henikoff and L. Shapiro. Interesting patterns for model–based matching. In *ICCV*, 1990.

[16] P. Horaud and R.C. Bolles. 3DPO: A system for matching 3-D objects in range data. In A.P. Pentland, editor, *From Pixels to Predicates*, pages 359–370. Ablex Publishing Corporation, Norwood, New Jersey, 1986.

[17] K. Ikeuchi. Generating an interpretation tree from a CAD model for 3D-Object recognition in bin-picking tasks. *Int. J. Comp. Vision*, 1(2):145–165, 1987.

[18] Richard E. Korf. Search: A survey of recent results. In Howard E. Shrobe and The American Association for Artificial Intelligence, editors, *Exploring Artificial Intelligence*, chapter 6, pages 197–237. Morgan Kaufmann Publishers, Inc., 1988.

[19] P. P. Loutrel. A solution to the hidden-line problem for computer-drawn polyhedra. *IEEE Trans. on Computers*, 19(3):205–210, March 1970.

[20] D. G. Lowe. Three–dimensional object recognition from single two–dimensional images. *Artificial Intelligence*, 31:355–395, 1987.

[21] H. Lu and L. G. Shapiro. Model–based vision using relational summaries. In *SPIE Conference on Applications of Artificial Intelligence VII*, March 1989.

[22] J. Ponce and D. Chelberg. Finding the limbs and cusps of generalized cylinders. *Int. J. Comp. Vision*, April 1987.

[23] F. P. Preparata and M. I. Shamos. *Computational Geometry: An Introduction.* Springer-Verlag New York Inc., 1985.

[24] A. Rosenfeld, R. A. Hummel, and S. W. Zucker. Scene labeling by relaxation operations. *IEEE Trans. Syst. Man Cybern.*, SMC-06, June 1976.

[25] A. Sanfeliu and K. S. Fu. A distance measure between attributed relational graphs for pattern recognition. *IEEE Transactions on Systems, MAn and Cybernetics*, SMC–13(13):353–362, May 1983.

[26] L.G. Shapiro and R.M. Haralick. Structural descriptions and inexact matching. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, PAMI–3(5):504–519, September 1981.

[27] L.G. Shapiro and R.M. Haralick. A hierarchical relational model for automated inspection tasks. In *Proc. 1st IEEE Int. Conf. on Robotics*, Atlanta, March 1984.

[28] L.G. Shapiro and R.M. Haralick. A metric for comparing relational descriptions. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, PAMI–7, 1985.

[29] I. E. Sutherland, R. F. Sproull, and R. A. Schumacker. A characterization of ten hidden-surface algorithms. *Computing Surveys*, 6(1), March 1974.

[30] J. R. Ullman. An algorithm for subgraph homomorphisms. *J. Assoc. Comput. Mach.*, 23:31–42, January 1976.

[31] J. T. Jr. Welch. A mechanical analysis of the cyclic structure of undirected linear graphs. *Journal of the Association for Computing Machinery*, 3(2):205–210, April 1966.

[32] Seungku Yi. *Illumination Control Expert for Machine Vision: A Goal Driven Approach.* PhD thesis, Department of Electrical Engineering, University of Washington, Seattle, Washington, 1990.