Three-Dimensional Object Recognition Systems
A.K. Jain and P.J. Flynn (Editors)

Image Prediction for Computer Vision

Octavia I. Camps [a] , Linda G. Shapiro [b] , and Robert M. Haralick [c]

[a] Dept. of Electrical and Computer Engineering
The Pennsylvania State University
University Park, Pennsylvania 16802 USA

[b] Dept. of Computer Science and Engineering, FR-35
University of Washington
Seattle, Washington 98195 USA

[c] Dept. of Electrical Engineering, FT-10
University of Washington
Seattle, Washington 98195 USA

Abstract

To recognize objects and to determine their poses in a scene we need to find correspondences between the features extracted from the image and those of the object models. Knowledge of the degree to which image features might break up or disappear under different lighting and viewing conditions is essential to automate the design of computer vision systems that can work with problems of practical complexity. In this paper we describe how the model-based vision system **PREMIO** (PREdiction in Matching Images to Objects) models some of the physical processes involved in the image formation and feature detection processes to predict and evaluate the features that can be expected to be detected in an image of an object in a semi-controlled environment. We also illustrate how these predictions can be used to successfully control the inherent combinatorial explosion of the relational matching approach commonly used in object recognition.

1. Introduction

To recognize objects and to determine their poses in a scene we need to find correspondences between the features extracted from the image and those of the object models. Most feature–based matching schemes assume that all the features that are potentially visible in a view of the object will appear with equal probability. The resultant matching algorithms have to allow for "errors" without really understanding what they mean, and usually get lost in the high combinatorics of the problem [12]. Thus, the application of these matching algorithms has been reduced to very simple tasks where only very few features are needed. Therefore the knowledge of the degree to which each feature might break up or disappear under different lighting and viewing conditions is essential to the design of computer vision systems that can work with problems of practical complexity. In this paper we

describe how a new model-based vision system, **PREMIO** (PREdiction in Matching Images to Objects), models some of the physical processes involved in the image formation and feature detection processes to predict and evaluate the features that can be expected to be detected in an image of an object in a semi-controlled environment. We also illustrate how these predictions can be used to successfully control the inherent combinatorial explosion of the relational matching approach commonly used in object recognition.

2. PREMIO Overview

PREMIO uses CAD models, surface reflectance properties, light sources, sensor 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 will be used to automatically generate vision algorithms. The system is illustrated in Fig. 1. Our 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.

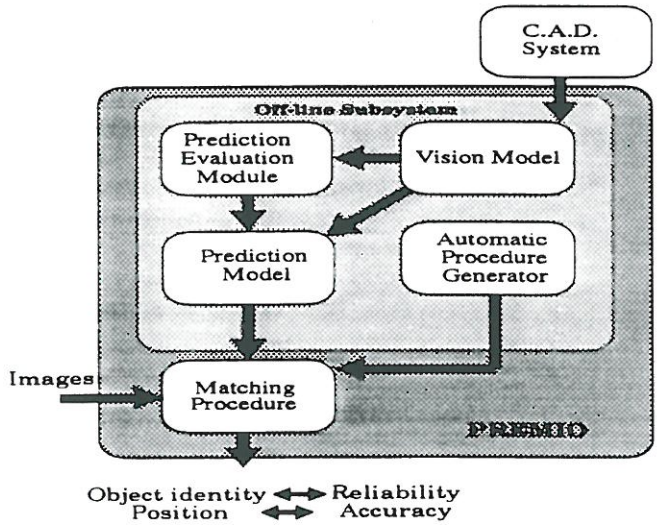


Figure 1. PREMIO: A Model-Based Vision System

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 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, taken from a given viewpoint and under a given light source and sensor configuration. The output of the Prediction Module is organized as the Prediction Model. The automatic procedure generator takes as its input the Prediction Model and creates the matching procedure to be used for matching the image features against the object models. The online subsystem consists of the matching procedure generated by the offline subsystem. It uses the Vision Model, the 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.

3. The Vision Model

A representation is a set of conventions about how to describe entities [25]. A *description* makes use of the conventions of a representation to describe some particular thing [25]. Finding an appropriate representation is a major part of any system-design effort, and in particular in the design of a machine vision system.

The vision model in a machine vision system is a *representation* of the world in which the system works. The entities that must be described by the representation of the world are the objects to be imaged and the characteristics that these images will have. In this paper we will limit ourselves to images that are formed when light is reflected off the surface of an object, such as photographs, but the concepts discussed here can be easily extended to other types of images such as range or X-ray images. The characteristics of an object image obtained by the usual optical means depends on several factors: 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 on the characteristics of the processes that "observe" the image.

Following the previous discussion, we can divide the problem of designing the vision model into three subproblems: how to describe the objects, how to describe the light sources and sensors, and how to describe the processes that observe the images. Fig. 2 shows a block diagram of the PREMIO vision model.

3.1. The Object Model

An image of an object is a two-dimensional pattern of brightness. How this pattern is produced depends not only on the geometry of the object but also on the way that light interacts with the object surface and the sensors. Hence, we have divided the object model into two submodels: the topological object model and the physical object model. The topological object model describes the geometry of the objects in PREMIO's world. The physical object model describes how the light interacts with the object surfaces and the sensors according to the laws of physics.

3.1.1. The Topological Object Model

The topological object model describes the geometric characteristics of the objects in PREMIO's world, and the relations among their faces, edges and vertices. In many domains, all the possible objects to be recognized are known and can be, if they are not already, modeled using a CAD system. Hence, PREMIO assumes it

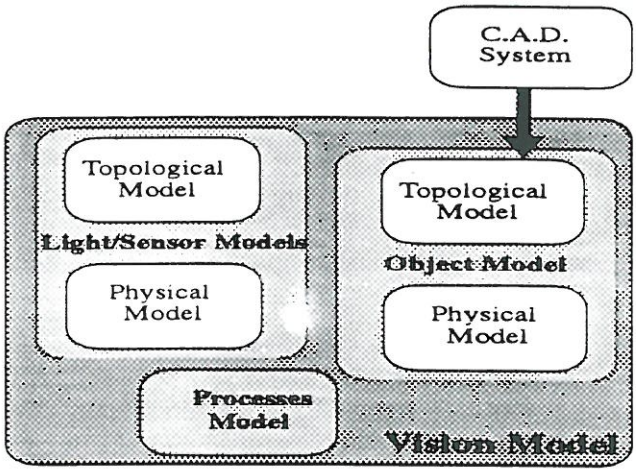


Figure 2. PREMIO's vision model.

has available CAD models of all the objects. A CAD system describes the geometry of the objects that it models, but not necessarily in the most adequate form for our application. CAD systems model 3D-objects and their component parts, providing view independent descriptions and interactive manipulation capabilities to observe and combine the different parts that can be automatically machined.

There are a number of commercial and experimental systems available for geometric modeling. A complete survey can be found in [21]. PREMIO uses the geometric modeler PADL2, designed by H. B. Voelcker and A. G. Requicha at the University of Rochester and distributed by Cornell University. This system is a constructive solid geometry (CSG) modeler, but has the ability to convert the CSG representation of any object to its boundary representation (BREP). Its primitives are: spheres, cylinders, cones, rectangular parallelepipeds, wedges and tori. Its main advantages are its capacity for fast wireframe drawing of the objects being designed that makes it very friendly to the user and its ability to convert any CSG representation to its boundary representation.

PREMIO's topological object model is a hierarchical, relational model similar to the one proposed by Shapiro and Haralick [22]. The object model is called a *topological object model* because it represents not only the geometry of the objects but also the relations among their faces, edges, and vertices. This information is redundant in the sense that it can be derived from the geometrical information, but it is included to speed up the system.

The model has five levels: a world level, an object level, a face level, a surface/boundary level, and an arc level. Fig. 3 shows a diagram of the topological object model. The world level at the top of the hierarchy is concerned with the arrangement of the different objects in the world . The object level is concerned

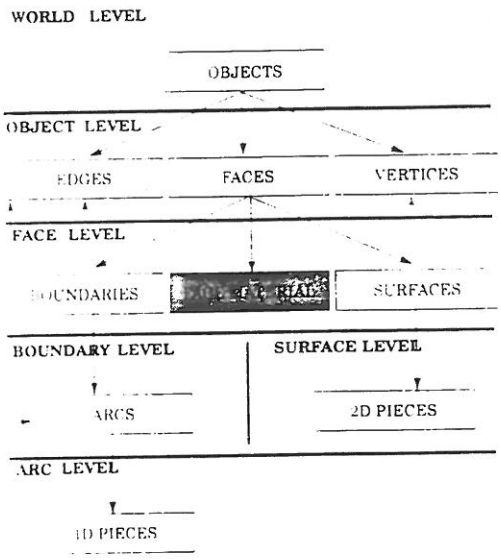


Figure 3. Topological Object Model Structure.

with the arrangement of the different faces, edges and vertices that form the objects. The face level describes a face in terms of its surfaces and its boundaries. The surface/boundary level specifies the elemental pieces that form those surfaces and the arcs that form the boundaries. Finally, the arc level specifies the elemental pieces that form the arcs.

To create the topological object model from the PADL2 model, we use 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, topology and vertex information that we need. To find the boundaries of a face, its arcs must be grouped together to form closed loops. These loops are found using an algorithm developed by Welch [24] to find closed loops in an undirected graph. At the same time that the boundary information is obtained, 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.

3.1.2. The Physical Object Model

Given the physical properties of a material it is possible to predict the properties of images of this material. PREMIO works with images produced by ordinary optical processes like photography or CCD cameras.

Buchanan [7] showed that the reflectance model presented by Cook and Torrance [10] is the most accurate when the light is completely unpolarized. However, polarized light is commonly used in vision tasks to suppress specular reflections from metal surfaces [3, 6]. Furthermore, monochromatic light is always polarized. Yi *et al.*'s proposed a light reflection model that is an extension of Cook and Torrance's

model, but that incorporates the reflection effects [13] found when the incident light is partially or totally polarized, and it is the one used in PREMIO.

In Yi *et al.*'s model the light sources are assumed to be dimensionless point light sources that radiate in every direction. Let P be a point on a surface, N be the surface normal at P, and I_l^i be the incident light from light l. The reflection models predict the intensity of the reflected light at P, I^r, as the sum of two terms. The first term is the ambient component, and the second term contains the specular and diffuse components summed over the number of lights present:

$$I^r = I_a^i R_a f + \sum_l I_l^i (N \cdot L_l) dw_l^i (s R_s + d R_d) \qquad (1)$$

where I^r is the intensity of the reflected light, I_a^i is the intensity of the incident ambient light, R_a is the ambient reflectance, f is the unblocked fraction of the hemisphere, I_l^i is the average intensity of the incident light l, N is the unit surface normal, L_l is the unit vector in the direction of the light l, dw_l^i is the solid angle of a beam of the incident light l, s is the fraction of reflectance that is specular, R_s is the specular bi-directional reflectance, d is the fraction of reflectance that is diffuse and R_d is the diffuse specular reflectance.

The model calculates the specular component by representing the object surface as a collection of many small planes which are called microfacets. The spectral reflectance function R_s is:

$$R_s = \frac{FDG}{\pi (N \cdot L)(N \cdot V)}$$

where F is the Fresnel term, D is the microfacet distribution, G is the geometric attenuation, N is the surface normal, L is the unit vector in the light direction and V is the unit vector in the viewer direction. Then

$$F = \rho_{\parallel} \cos^2 + \rho_{\perp} \sin^2$$

with

$$\tan^2 = \frac{I^r}{I_l^i} \qquad \cos = N \cdot L$$

$$\rho_{\perp} = \frac{c^2 + b^2 - 2a \cos + \cos^2}{c^2 + b^2 + 2a \cos + \cos^2} \qquad \rho_{\parallel} = \rho_{\perp} \frac{c^2 + b^2 - 2a \sin \tan + \sin^2 \cos^2}{c^2 + b^2 + 2a \sin \tan + \sin^2 \cos^2}$$

$$a = \sqrt{\frac{\sqrt{c^2 + 4n^2k^2} + c^2}{2}} \qquad b = \sqrt{\frac{\sqrt{c^2 + 4n^2k^2} - c^2}{2}}$$

$$c = n^2 - k^2 - \sin^2$$

where n is the index of refraction and k is the extinction coefficient of the material; and

$$D = \frac{\exp^{-((\tan \delta)/m)^2}}{m^2 \cos^4 \delta}$$

where m is the root mean square slope of the facets, and δ is the angle between the surface normal N and the angular bisector between L and V, H. The parameter m

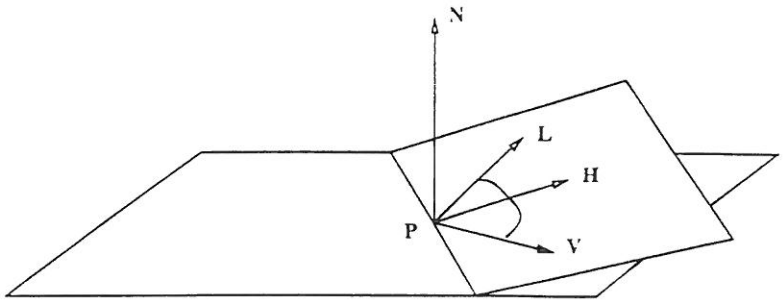


Figure 4. Light reflection on a microfacet.

is a scalar between 0 and 1 and is a measure of the roughness of the surface: the rougher the surface, the greater the value of m;

$$G = \min\left\{1, \frac{2(N \cdot H)(N \cdot V)}{(V \cdot H)}, \frac{2(N \cdot H)(N \cdot L)}{(V \cdot H)}\right\}$$

The diffuse component is modeled in both models as a constant denoted R_d. The diffuse component is independent of the angle of illumination, and hence it is taken to be equal to the specular reflection with the illumination on the surface normal ($L = V = N = H$).

In summary, the physical object model consists of a set of constants for each of the object surfaces such that they completely specify the parameters needed in Yi *et al.*'s reflection model. These constants are:

R_a	ambient reflectance	R_d	diffuse reflectance
s	specular fraction	d	diffuse fraction
m	root mean square slope of microfacets	n	index of refraction
k	extinction coefficient		

3.2. The Light Sources and Sensors Model

Image formation occurs when a sensor registers radiation that has interacted with physical objects. The quality of an image is greatly affected by the sensor resolution and the scene illumination. In most industrial applications the imaging acquisition setup is known and controllable. That is, the type of light sources and sensors used as well as their relative geometric arrangement are known, if not fixed. This knowledge is incorporated into the vision model of PREMIO.

As with the object model, the light sources and sensors model can be divided into a topological and a physical model. The topological model describes the possible geometric or space configurations of the light sources and sensors with respect to the object being imaged. The physical light sources and sensors model describes their physical characteristics: the light wavelength and polarized components, the response of the sensor to the radiation input, and its resolution.

3.2.1. The Light Sources and Sensors Topological Model

The light sources and sensors topological model consists of all possible geometrical arrangements of the light sources and sensors with respect to the object reference frame. These configurations could be just a few or hundreds depending on the object and the particular application. In PREMIO the light sources and sensors are placed on spheres centered at the origin of the object coordinate system, called the *illumination* and *viewing* spheres. The radii of these spheres are large enough to contain the whole object. The points at the illumination and viewing spheres constitute a continuous space which is sampled uniformly.

3.2.2. The Light Sources and Sensors Physical Model

The light sources and sensors physical model describes properties and characteristics of the light sources and sensors that respond to the laws of physics and that are independent of their position in space. The light source physical model describes light characteristics such as the polarization and the wavelength distribution of the sources. The sensor physical model describes the magnification of the sensors and their transfer characteristics.

The Light Sources Physical Model

In PREMIO, the light sources are monochromatic point sources that irradiate partially or totally polarized light in all directions. Hence, the physical model of the light sources consists of the intensity values of the parallel and perpendicularly polarized components of the sources.

The Sensor Physical Model

The sensor physical model describes how images are formed. A gray-scale image is a two-dimensional pattern of brightness. To understand how this pattern is formed, we need to answer two questions: (1) *What determines where, on the image, some point on the object will appear?* and (2) *What determines how bright the image of some surface on the object will be?* The first question can be answered by making a first-order approximation of the camera as a "pinhole" camera. Using this approximation, images result from projecting scene points through a single point, the center of projection, onto an image plane at a distance f in front of the camera (See Fig. 5). Given a coordinate system, called the *camera coordinate system*, with its $z-axis$ parallel to the optic axis of the camera lens and such that the camera lens is at (x_0, y_0, z_0) and the image plane has equation $z = f + z_0$, the coordinates $(x', y', f + z_0)$ of the projection of a point with coordinates (x, y, z) onto the image plane are given by:

$$\begin{aligned} x' &= x^*/t^* \\ y' &= y^*/t^* \end{aligned} \tag{2}$$

$$\begin{pmatrix} x^* \\ y^* \\ t^* \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & -x_0 \\ 0 & 1 & 0 & -y_0 \\ 0 & 0 & 1/f & -z_0/f \end{pmatrix} \cdot \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} \tag{3}$$

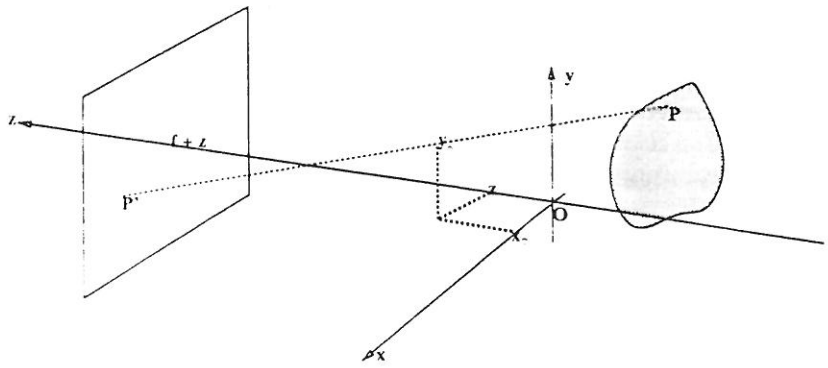


Figure 5. Imaging Geometry: Perspective Projection.

Next, we answer the second question of what determines how bright the image of some surface on the object will be. Different points on the object in front of the imaging system will have different intensity values on the image depending on the amount of the incident radiance, how they are illuminated, how they reflect light, how the reflected light is collected by the lens system of the imaging system, and how the sensor responds to the incoming light. The image intensity I is proportional to the scene radiance. The scene radiance, the amount of light emitted by the surface, depends on: (1) the amount of light falling on the surface; (2) the fraction of the incident light that is reflected; and (3) the geometry of the reflection, (*i.e.* the direction from which it is viewed as well as the direction from which it is illuminated). Mathematically, the image intensity of a given point P in a given object surface is given by [13]:

$$I = \int C S(\lambda) Q(\lambda) I^r(\lambda) d\lambda$$

where C is the lens collection factor, S is the wavelength dependent sensor responsivity, Q is the spectral distribution of the illumination source, λ is the wavelength of the light, and I^r is the reflected intensity at P given by equation 1. The lens collection factor, C, is the portion of the reflected light that comes through the lens system and affects the film. The lens collection C is given by [13]:

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

where f is the focal distance of the lens, a is the diameter of the lens, and α is the angle between the reflected 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 by 1, independent of the wavelength of the incident light.

The *digital images* with which computers deal are arrays of integers. Hence, both the image domain and the image range have to be sampled. Sampling the

domain involves the sampling interval and the pattern of the sampled points. The sampling interval determines how many sampled points (pixels) there are in the image. Television frames, for example, might be quantized into 450x560 pixels, while the resolution of general film is approximately 40 lines/mm, that is 1400x1400 pixels for a 35mm slide. The pattern into which the image range is sampled is called its tessellation. The tessellation pattern used in PREMIO, as with almost all machine vision systems, is a rectangular pattern. Sampling the range, corresponds to a *quatization* of the intensity values into a number of different *gray levels*. The number of levels determines the number of bits necessary to represent the intensity of a pixel. In PREMIO, images are quantized into 256 levels corresponding to 8-bit pixels.

In summary, a sensor physical model in PREMIO is specified by giving the following constants:

f camera focal length	a diameter of the camera lens
w film width	h film height
R camera resolution	

3.3. The Processing Algorithms Model

An image is only as good as the processes used to extract the information that it contains. In the context of machine vision, these processes are the algorithms that are used in the feature extraction process. These algorithms and their performance must be included in the vision model of the system. Examples of processing algorithms are edge detectors, corner detectors, line finders, etc. The processing algorithms model in PREMIO consists of a set of gradient edge operator masks, a set of threshold values (used to decide when the gradient is large enough to assume that there is a local edge at a particular pixel), a thinning algorithm, a linking algorithm and a corner detection algorithm[13].

4. Feature Prediction Module

Given a vision model representing the world, the goal of the prediction module is threefold: (1) it must predict the features that will appear in an image taken of the object from a given viewpoint and under given lighting conditions; (2) it must evaluate the detectability of the predicted features; and (3) it must organize the data produced by (1) and (2) in an efficient and convenient way for later use. The visible features can be edges, corners, holes, or any complex relationship among these primitive features. The *detectability* of a feature for a given sensor and detector is the probability of finding the feature using that sensor. Additionally, the prediction module should evaluate the reliability and accuracy of the predicted features. The *reliability* of a feature is the probability of correctly matching the detected feature to the corresponding one in the model. The *accuracy* of a feature is a measure of the error or uncertainty propagated from the detected feature to a geometric property like the pose of the object.

4.1. Predicting Features

Given a three-dimensional object and the imaging geometry model, we want to determine which edges and surfaces are visible on the image of the object. 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. Realistic image synthesis is a computer graphics area that has as its ultimate goal to produce synthetic images as realistic as photographs of real environments. Amanatides [1] surveyed different techniques used in realistic image generation. In general there is a tradeoff between processing time and realism. A particularly powerful technique used to achieve realism is ray casting: we cast a ray from the center of projection through each picture element and identify the visible surface as the surface intersecting the ray closest to the center of projection. Bhanu et al. [4] use ray casting to generate range images for their vision model. Other systems, [11, 15] use a different technique, called polygon mesh shading. This technique approximates the objects with polygon meshes, reducing considerably the required processing time while maintaining an appealing realism.

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.

4.1.1. 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 edges 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 line segments, triplets, corners, forks, holes, etc. A block diagram of the feature prediction module and its connections with the vision model is given in Fig. 6. Fig. 7 illustrates the model-based feature prediction process. Fig. 7(a) shows a raycasted image of an object. Fig. 7(b), (c) and (d) show the results of each of the steps.

Ponce and Chelberg [19] used this approach to predict features for generalized cylinder objects. However, they assumed an imaging system with orthographic projection, and they did not consider the effects of lighting or material properties.

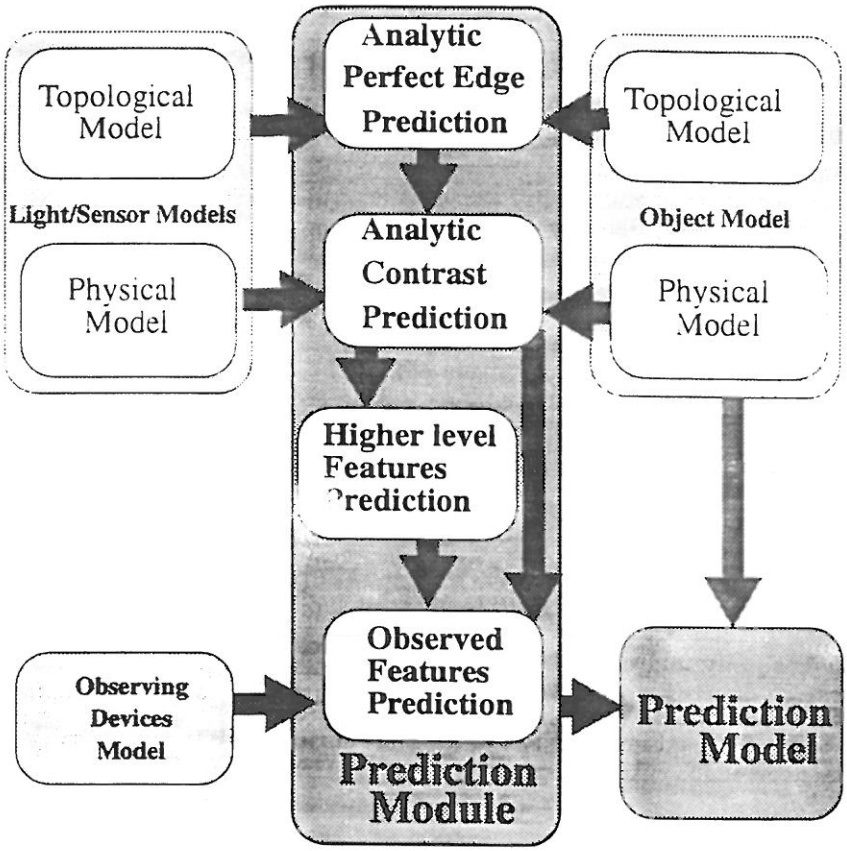


Figure 6. PREMIO's Feature Prediction Module.

4.1.2. Analytic 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 [23]. 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 representation we need to introduce the following definitions [17]:

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.

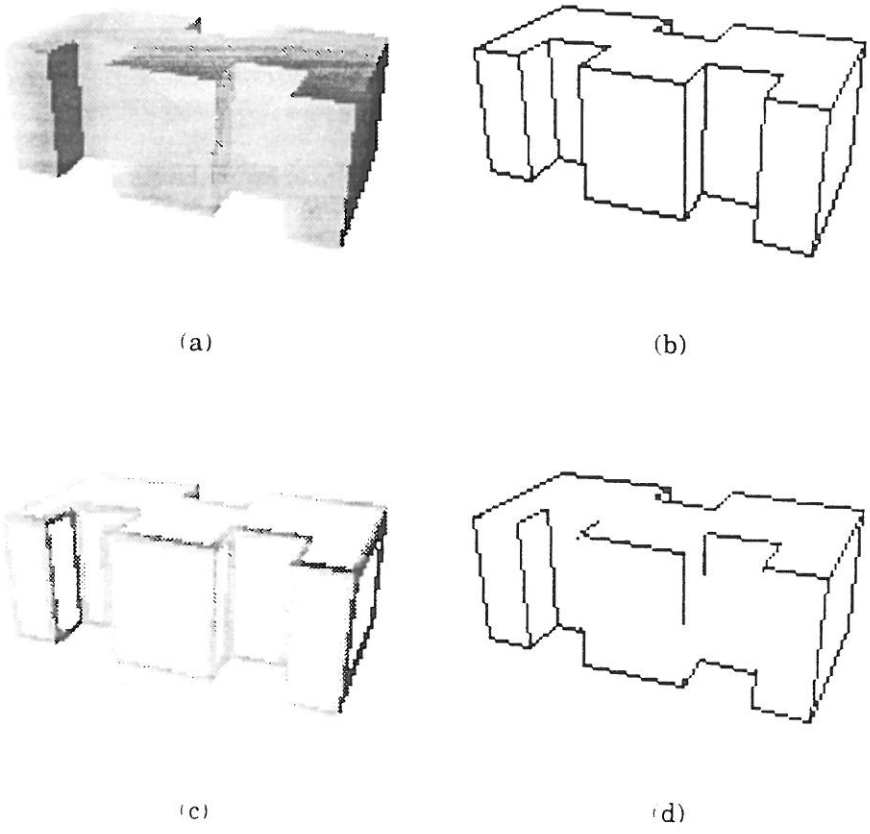


Figure 7. Model-Based Feature Prediction. (a) Raycasted Object. (b) Wireframe. (c) Intensity image along the edges. (d) Edges after applying an edge operator.

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 we have to take the following steps: (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) find for each cusp and T-junction point the object surface closest to the point of view.

Finding Limbs and Cusp Points

To find the analytical expressions for the limbs and cusp points we use an approach similar to the one used by Ponce and Chelberg [19], 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 with implicit equation $f(X, Y, Z) = 0$. Then, the vector of sight $\vec{v}$ from P_0 to P is given by:

$$\vec{v} = (X - X_0, Y - Y_0, Z - Z_0) \tag{4}$$

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

$$\vec{N} = \left(\frac{\partial f}{\partial X}, \frac{\partial f}{\partial Y}, \frac{\partial f}{\partial Z} \right). \tag{5}$$

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:

$$\begin{cases} \vec{v} \cdot \vec{N} = 0 \\ f(X, Y, Z) = 0 \end{cases} \tag{6}$$

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

$$\begin{cases} X = X(t) \\ Y = Y(t) \\ Z = Z(t) \end{cases} \tag{7}$$

with $t_{\min} \leq t \leq t_{\max}$. Then, the tangent vector $\vec{T}$ to the limb is given by:

$$\vec{T} = \left(\frac{\partial X}{\partial t}, \frac{\partial Y}{\partial t}, \frac{\partial Z}{\partial t} \right) \tag{8}$$

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:

$$\begin{cases} \vec{T} \cdot \vec{v} = 0 \\ X = X(t) \\ Y = Y(t) \\ Z = Z(t) \end{cases} \tag{9}$$

with $t_{\min} \leq t \leq t_{\max}$.

The process of finding the limbs 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 we need to project the limbs and boundaries of the object onto the image plane; to find the T-junctions we need to intersect the resulting contours. The intersection detection problem for n planar objects has been extensively studied and it can be solved in $O(n \log n + s)$ time [20], where s is the number of intersections. In our case, the objects are the set of contours. 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 solution for the T-junctions between contours that result from projecting straight lines and conics [8]. To find the other 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], and Loutrel [17], 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 $O(f + (2 \times 3 \times e))$.

4.1.3. Using Material and Lighting Knowledge

Once the wireframe of the object with the hidden lines removed has been obtained, the prediction module uses its knowledge about the sensor, the reflectance properties of the surfaces material, and the lighting conditions to predict the intensity of the reflected light in the neighborhood of the wireframe contours.

While the contours obtained in the previous stage are continuous, the digital

images with which computers deal are discrete. Hence, the continuous contours have to be discretized according to the resolution of the sensor being modeled. This is accomplished by using the well known graphics algorithm due to Bresenham [5]. The intensity of the reflected light is computed in a neighborhood of the contour pixels by applying the reflectance surface model and the light model to the selected pixels. Furthermore, each of these pixels has associated one or more proximate contours and their corresponding 3D boundary or limb.

4.1.4. Using the Processing Algorithms Model

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 characteristics. The prediction module uses the processing algorithms model described in section 3 to predict which features can be detected. The module applies the modeled algorithms to the intensity pixels predicted in the previous stage while keeping track of their associated 3D features.

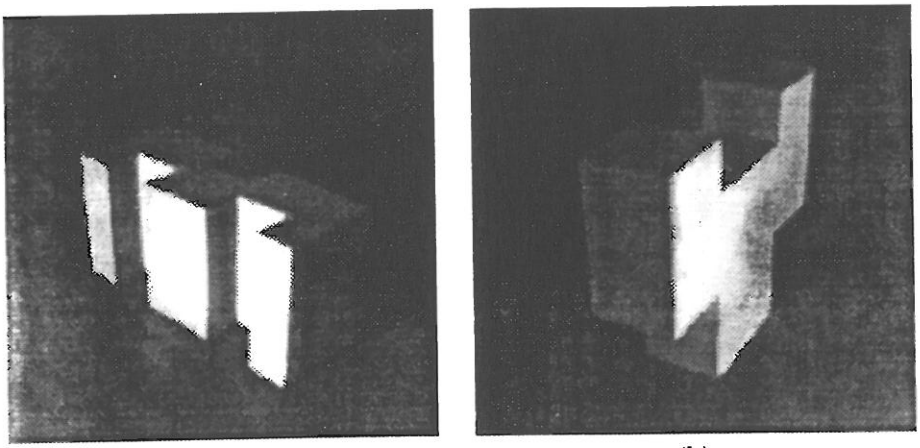
Simple features such as edges can be interpreted by themselves, or can be grouped to be considered as higher-level features. Matching *perceptual groupings* of features was suggested first by Lowe [18]. Henikoff and Shapiro [14] proposed using arrangements of triplets of line segments, called *interesting patterns*. Other useful high-level features are junctions and closed loops. In general a higher-level feature will be more useful than a lower-level feature to recognize and locate an object. Of course, there is a tradeoff between the amount of information that a feature represents and the cost of extracting that feature from the image.

4.2. Evaluating Predicted Features

After a feature is predicted, its potential utility must be evaluated. In order for a predicted feature to be useful, it has to be detectable in the image. For a given sensor and detector, the *detectability* of a feature is defined as the probability of finding the feature using that detector on an image taken with that sensor. We estimate the detectability of a feature by the frequency with which it shows up in a prediction, when the light and sensor positions are varied over the illumination and viewing spheres.

4.3. Output of the Predictor Module

For a given object, a configuration of light sources, and one or more sensors, the output of the predictor module is a hierarchical relational data structure similar to the one defined in section 2. This structure will be called a *prediction* of the object. Each prediction contains a set of image features, their attribute values such as detectability, and their corresponding three-dimensional features. The prediction also has five levels: the image level, an object level, a region level, a boundary level, and an arc level. The image level at the top of the hierarchy is concerned with the imaging conditions that generated the prediction, the general object position, the background information, etc. The object level is concerned with the different regions, edges and junctions that will appear on the image. The region level describes the regions in terms of their boundaries. The boundary level



(a) (b)

Figure 8. (a) Cube3Cut. (b) Fork.

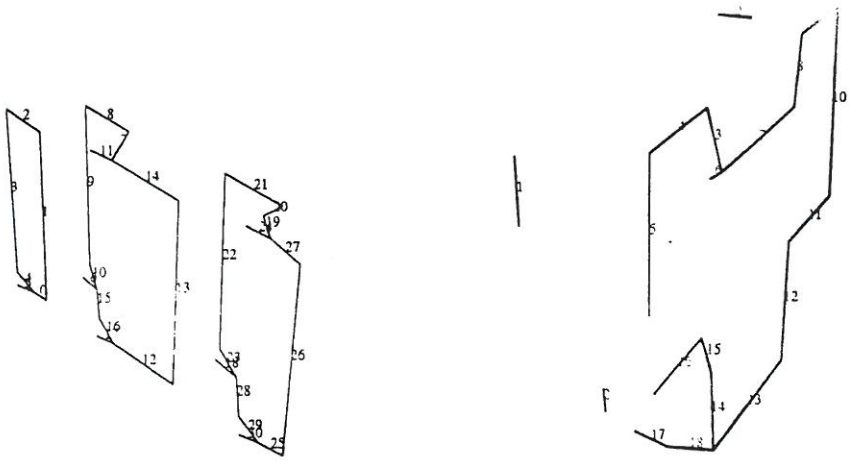
specifies the arcs that form the boundaries of the regions. Finally, the arc level specifies the elemental pieces that form the arcs.

5. Illustrative Examples

In this section we will show the usefulness of the predictions obtained by PREMIO through some illustrative examples. Fig. 8 shows real images of *Cube3Cut* and *Fork*, two of the objects modeled in PREMIO. Fig. 9 shows the line segments obtained after processing the images with a sequence consisting of a Sobel edge operator, a connected shrinking, an edge linking, and a corner detection algorithms. As a result of the illumination conditions when the images were taken, and the use of "default" parameters in the image processing sequence, many segments are missing and others are broken.

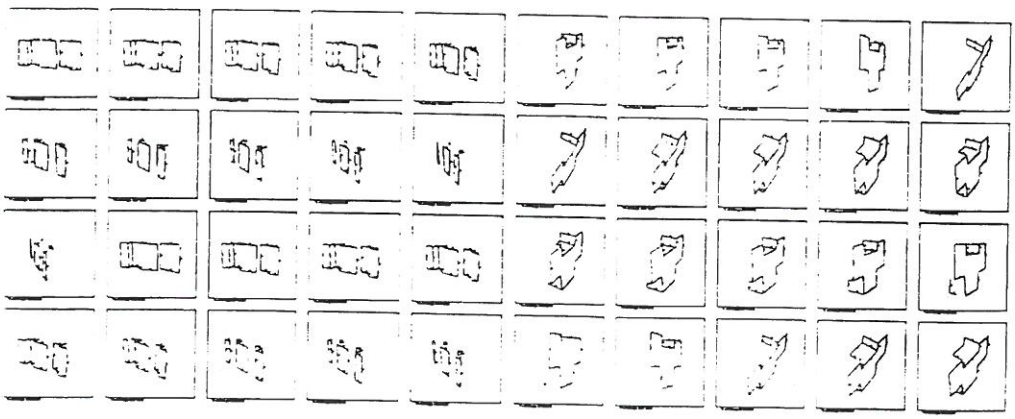
The results shown in Fig. 9 are not surprising if we examine the predictions produced by PREMIO for similar viewing and illumination conditions. Fig. 10 shows some of the predictions generated for views within the same view aspect of the ones shown in Fig. 8 and with the light at approximately the same location. Even though the predicted views are different from the real ones being considered, they show which segments are more prone to disappear or appear fragmented.

PREMIO summarizes hundreds of predictions of an object into a single model M. The model consists of a set of features (segments), a set of relational tuples of features (junctions and chains of segments), and an attribute mapping for the features (midpoint coordinates, length, and orientation statistics of the segments). The features and the relational tuples are ranked in decreasing order of detectability. Fig. 11 shows visualizations of the sets of features of the models of Cube3Cut



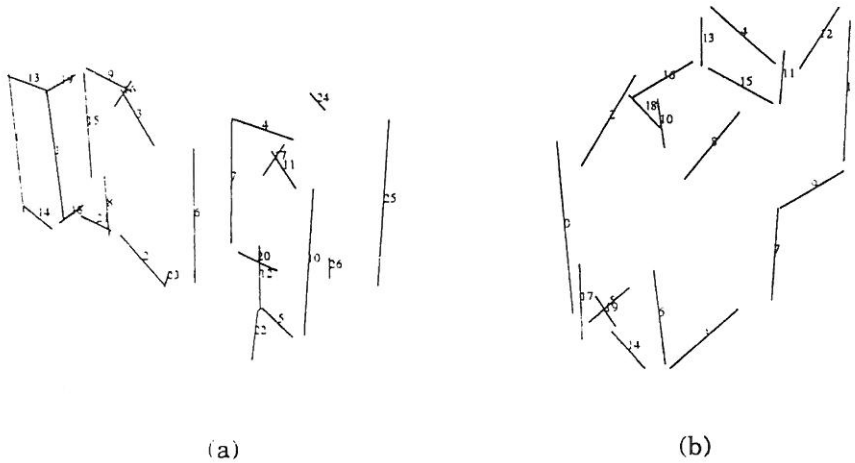
(a) (b)

Figure 9. (a) Cube3Cut segments. (b) Fork segments.



(a) (b)

Figure 10. (a) Cube3Cut: predicted images. (b) Fork: predicted images.



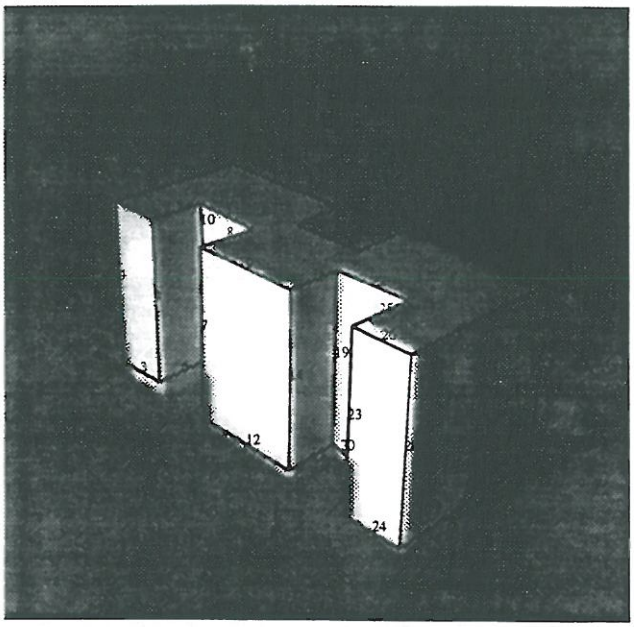
(a) (b)

Figure 11. Cube3Cut and Fork models (segments). (a) Cube3Cut. (b) Fork.

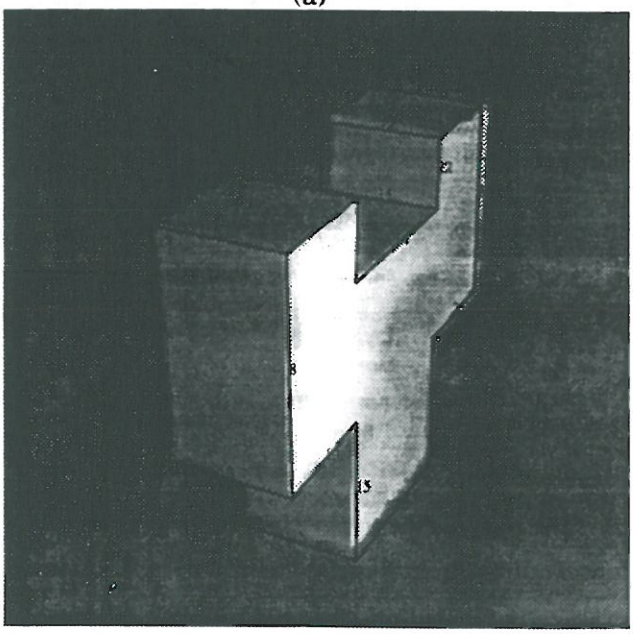
and Fork respectively. The features are drawn as segments with their mean attribute values. The numbers shown by the segments are the feature ID's indicating the relative detectability, with the lower the number, the higher the detectability.

An example of how these predictions can be used to reduce the combinatorial explosion of the relational matching problem can be found in [8, 9]. There, the relational matching problem is framed as the Bayesian problem of finding a set of correspondences h between features of a model M and features of an image I, such that the *a posteriori* probability $P(M, h|I)$ that the given image I is an observation of M under the transformation h is maximized. Furthermore, it is shown that this problem is equivalent to maximizing the joint probability $P(M, h, I)$, since the maximization does not involve I.

In order to solve this problem, we need first to estimate the probability distribution $P(M, h, I)$ and then find the set of correspondences h such that the probability is maximum. Given a model M, the predicted images produced by PREMIO are observations of the model, each one having associated a different set of correspondences. Thus, these predictions correspond to samples of the probability distribution $P(M, h, I)$ and they can be used to estimate the parameters of the probability distribution (See [8, 9].). A set of correspondences h that maximizes the $P(M, h, I)$ probability, can be found by using an iterative-deepening-A* (IDA*) search [16] in a tree where each node represents a model feature, the branches represent the possible image feature assignments, and a path from the root to a leaf represents a possible set of feature correspondences. The IDA* algorithm consists of a sequence of depth-first searches. It starts with an initial threshold value equal to the estimated probability for a path starting at the root. In each iteration, the algorithm is a pure depth-first search, cutting off any branch that has an estimated probability smaller than the current threshold value. If a solution is expanded, the algorithm is finished. Otherwise, a new threshold value is set to the maximum



(a)



(b)

Figure 12. Cube3Cut and Fork pose estimation. (a) Cube3Cut. (b) Fork model.

estimated probability that was below the previous threshold, and another depth-first search is begun from scratch. As in the well known A* heuristic search, if the estimated probability is an overestimate of the real probability, IDA* finds the optimal solution with time complexity equal to an exponential function of the error of the estimate. The advantage of IDA* over A* is that since each iteration of the algorithm is a depth-first search, the memory complexity is a linear function of the depth of the solution, instead of being exponential. The number of nodes opened by IDA* is asymptotically the number of nodes opened by A*, provided that the tree grows exponentially. In practice, IDA* runs faster than A*, since its overhead per node is less than the overhead for A*.

It was found that the use of the predictions resulted in a pruning ratio, defined as the percentage of pruned paths relative to the total number of opened paths in the tree, between 40% and 60%.

Fig. 12 shows the estimated wireframes obtained by using 11 and 10 feature correspondences superimposed on the images of Cube3Cut and Fork respectively. For Cube3Cut, only 51 paths were opened and 29 of these were pruned, resulting on a pruning ratio of 56.86% and an execution time of 1.75 sec in a SPARC station 2. For Fork, 196 paths were opened and 126 of them were pruned, resulting in a pruning ratio of 64.29% and an execution time of 2.9 sec.

6. Conclusion

The CAD-based vision system PREMIO combines techniques of analytic graphics, CAD models of 3D objects and knowledge of surface reflectance properties, light sources, sensors characteristics, and the performance of feature detectors to predict and evaluate the features that can be expected to be detected in an image. The predictions generated in this way are powerful tools for object recognition. These predictions can be used to dramatically reduce the search space in a feature-based object recognition algorithm.

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. Batchelor, D. Hill, and D. Hodgson. *Automated Visual Inspection*. IFS Publications Ltd., Bedford, UK, 1985.

4 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.

5 J. Bresenham. Algorithm for computer control of digital plotter. *IBM Syst. J.*, 4(1):25–30, 1965.

6 A. Browne and L. Norton-Wayne. *Vision and Information Procesing for Automation*. Plenum Press, New York, N.Y., 1986.

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

8 O. I. Camps. *PREMIO: The Use of Prediction in a CAD-Model-Based Vision System*. PhD thesis, Department of Electrical Engineering, University of Washington, Seattle, Washington, 1992.

9 O. I. Camps, L. G. Shapiro, and R. M. Haralick. Object Recognition Using Prediction and Probabilistic Matching. In *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1044–1052, Raleigh, North Carolina, July 1992.

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

11 P. J. Flynn. *CAD-Based Co[illegible]puter Vision: Modeling and Recognition Strategies*. PhD thesis, Michigan State University, 1990.

12 W. E. L. Grimson. The combinatorics of object recognition in cluttered environments using constrained search. In *Proc. of the International Conference on Computer Vision*, pages 218–227, 1988.

13 R. Haralick and L. Shapiro. *Computer and Robot Vision*. Addison-Wesley, 1992.

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

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

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

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

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

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

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

21 A. A. G. Requicha and H. B. Voelcker. Solid modeling: A historical summary and contemporary assesment. *IEEE Computer Graphics and Applications*, pages 9–24, March 1982.

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

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

24 J. T. J. 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.

25 P. Winston. *Artificial Intelligence*. Addison-Wesley Publishing Company, second edition, July 1984.