# Correspondence

## Efficient Data Structures for Model-Based 3-D Object Recognition and Localization from Range Images

Wu Wang and S. S. Iyengar

*Abstract*—**This paper defines and investigates a fundamental problem in computer vision: recognition and localization of multiple free-form 3-D objects in range images. This facility is important in an automated manufacturing environment in the industry. *The emphasis throughout this paper will be on the design of efficient data structures and algorithms.* The principal results of our work are as follows:**

- **Surfaces are characterized by surface curvatures derived from geometric models of objects. Surface shapes and a knowledge representation scheme are uniquely defined and used in the search for corresponding surfaces of an object, based on an ordered feature space.**
- **Knowledge about model surface shapes is automatically abstracted from CAD models, and these models are also used directly in the vision process.**
- **Our technique will recognize objects by hypothesizing and locating them. Knowledge about object surface shapes is used to infer hypotheses, and CAD models are used to locate objects.**
- **One of the most important problems in 3-D machine vision is the recognition of objects from their partial view due to occlusion. Our approach is surface based and is not sensitive to noise or occlusion.**
- **A test system called free-form object recognition and localization (FORL) was implemented and tested on synthetic images.**

*Index Terms*—**CAD-based vision, geometric modeling, image processing, model-based machine vision, range image understanding, surface curvature, surface shape, 3-D object recognition and localization.**

### I. Introduction

In recent years, there has been a tremendous spurt in the recognition of 3-D objects in range images. Visual data obtained from range sensors by a robot provides 3-D range information about objects directly. Interpretation of range data by a vision system has been one of the major problems of vision research, e.g., ability to derive properties, such as extracting features and recognizing objects. Toward this objective, we develop an efficient approach for the recognition and localization of 3-D freeform objects in range images using the properties of algebraic surfaces. Bolles and Horaud [3] present an approach for finding the configuration of objects from range data by matching preselected features. Their method, because it is edge based, is sensitive to occlusion and noise. Gunnarsson and Prinz [7] propose a method for localization of industrial parts using CAD models but handle a very limited rotation of the objects from the models. Vemuri and Aggarwal [14] propose a method for determining the orientation of an object from a range image by point correspondence, which may be sensitive to noise. Flynn and Jain [6] describe a method for constructing relational graphs from CAD models. These relational graphs can subsequently be used for object recognition. Grimson and Lozano-Perez [17] discuss use of sparse local measurements of positions and surface normals to recognize and locate objects. Wong *et al.* [15] propose an attribute hypergraph-based object recognition. Bhanu and Nuttal [19] propose characterization of surface curvatures on curvature graphs to recognize objects.

This paper is organized as follows. In Section II, we explain one of the most common CAD model representations—boundary representation of solids. The invariant properties of surfaces are shown in Section III. We discuss the considerations for knowledge representation in intelligent systems and introduce our data structures in Section IV. The control strategy of the recognition and localization process is discussed in detail in Section V. The implementation details are shown in Section VI. Finally, we present our conclusions in Section VII.

#### A. Models for Machine Vision

Model representation has a significant effect on model-based object recognition. Many modeling methods have been proposed and used in computer vision systems. Wireframe models to describe solid object edges are commonly used in blockworld vision [12]. The approach we introduce is suitable for using either CAD models or automatically generated models to recognize and localize objects. In fact, automatic models are the same as CAD models in the sense of their data structures.

#### B. Computer-Aided Design Models

CAD models contain details about solid objects. A CAD system is generally used to design new shapes for automatic manufacture. It provides an interactive design interface, which is usually user friendly, and helps create, modify, and analyze a design. CAD models are very stable in representing geometric features of 3-D objects and revealing model structures in detail. Another reason for CAD model-based vision is the wide availability of CAD/CAM systems in industry. CAD models are perfect for producing images from models. However, machine vision, as a reverse process, does not easily use CAD models directly. We have developed an approach to use CAD models to generate hypothesis images for verification. Our approach uses the knowledge about surface shapes of objects to perform recognition reasoning. Knowledge about surface shapes of objects is abstracted from CAD models automatically.

There exist many schemes for representing solids. Among them, the most popular schemes are boundary representations (B-rep), constructive solid geometry (CSG), sweep representations, cell decompositions, spatial occupancy enumeration, primitive instances, and analytic solid modeling (ASM) [4], [5].

#### C. Boundary Representation of Solids

A boundary representation of a solid object $m$ can be defined as a set $m = f_1, f_2, \ldots, f_{p_m}$, where $f_i$, $1 \leq i \leq p_m$ is a surface patch. There are many conventional functions to describe surface patches in computer graphics, although there is no unique B-rep for a solid.

Methods of designing surface patches in CAD systems often use a set of discrete points called control points to help define surface patches. B-spline and Bezier surfaces are some commonly used functions.

Manuscript received January 10, 1991; revised January 29, 1992. This work was supported by a Department of Navy (1988) grant. Recommended for acceptance by Associate Editor J. Mundy.

W. Wang is with Metaphor Computer Systems, Mountain View, CA.

S. S. Iyengar is with the Department of Computer Science, Louisiana State University, Baton Rouge, LA 70803.

IEEE Log Number 9201776.

### D. Models Generated Automatically

The fundamental principle of automatically building object models consists of taking multiple views of an object, segmenting each such image, and dividing each image into a set of smooth segments that can be represented by a mathematical function. There are many different ways to represent surface segments. Most automatic model generation research concentrates on creating models in parametric function representation. There are segmentation methods [9] that describe surface segment regions by an explicit bivariate function $z = f(x, y)$. For instance, an image can be partitioned into smooth surface segments described by variable-order explicit bivariate functions. Explicit functions are not often used in automatic model generation research since they are not compatible with CAD models. However, our approach only requires models given in Section I-C. The $f_i$ in Section I-C can be represented in parametric functions, as CAD models, or in explicit functions. Therefore, our approach allows automatic generation of models for recognition and localization, which makes a self-contained vision system. On the other hand, if the same segmentation procedure is used to partition images in automatic model generation and segment the input image in recognition, the recognition and localization process will be more accurate and more efficient.

## II. Invariant Properties of Surfaces

If surfaces are recognized by their characteristics, object recognition can be reduced to a surface recognition problem—the so-called surface-based recognition. This is in contrast with the conventional edge-based recognition that recognizes objects by using edge characteristics and their relations. In order to recognize surfaces, we must have well-defined features or mathematical entities that can be used to distinguish between different entities of the same type. It is well known that curvature, torsion, and speed uniquely define the shape of 3-D curves [8], [10]. In the surface case, there are two basic mathematical entities that are considered in the analysis of smooth surfaces. They are referred to as the first and second fundamental forms of a surface. We will show how these forms uniquely characterize and quantify a general, smooth surface shape. Based on these fundamental forms, invariant surface characteristics, such as the Gaussian curvature and the mean curvature, are derived. These characteristics are *invariant* to changes in surface parameterization and to translation and rotations of object surfaces. A robust 3-D object recognition system should be view independent. Therefore, the use of invariant surface characteristics in 3-D vision systems is significant. Furthermore, the Gaussian curvature and the mean curvature are local surface properties that allow surface curvature to be used in occlusion situations.

### A. Surfaces

A curved surface can be defined as a polynomial in terms of two parameters $u$ and $v$. The surface $S$ is a set of points in 3-D space. The representation

$$\boldsymbol{p} = \boldsymbol{p}(u, v) = [x(u, v), y(u, v), z(u, v)] \tag{2.1}$$

is a mapping of an open set $U$ in the $uv$ plane onto $S$. If for all $(u, v)$ in $U$

$$J_x^2 + J_y^2 + J_z^2 \neq 0 \tag{2.2}$$

where $J_x^2$, $J_y^2$, and $J_z^2$ are Jacobians defined as

$$J_x = \frac{\partial(y, z)}{\partial(u, v)};\ J_y = \frac{\partial(x, z)}{\partial(u, v)};\ J_z = \frac{\partial(x, y)}{\partial(u, v)} \tag{2.3}$$

all derivatives of $\boldsymbol{p}(u, v)$ of up to order $m$ exist, and all such derivatives are continuous, then the curved surface defined by (2.1) is said to be of class $C^m$.

Condition (2.2) guarantees that the curved surface will not degenerate to a point or a curve and that it does not contain any singular points. This condition requires constraints on both the curve itself and the parameters. We will assume that the condition holds since all our geometric models are generated by piecewise-smooth surfaces, and the images are segmented into piecewise-smooth surfaces.

A parametric representation will be denoted by $\boldsymbol{p} = \boldsymbol{p}(u, v)$ and its partial derivatives by

$$\boldsymbol{p}_u = \frac{\partial \boldsymbol{p}}{\partial u},\ \boldsymbol{p}_v = \frac{\partial \boldsymbol{p}}{\partial v},\ \boldsymbol{p}_{uu} = \frac{\partial^2 \boldsymbol{p}}{\partial u^2},\ \boldsymbol{p}_{uv} = \frac{\partial^2 \boldsymbol{p}}{\partial v \partial u}.$$

If $\boldsymbol{p}$ is class $m \geq 2$, then $\boldsymbol{p}_{uv} = \boldsymbol{p}_{vu}$. Although that strictly speaking, a parametric representation (2.1) is a mapping, we will speak rather loosely and identify it with its image: a set of points $S$. Therefore, we say that $P$ is a point on $\boldsymbol{p} = \boldsymbol{p}(u, v)$ when $P$ is a point on the image of $\boldsymbol{p} = \boldsymbol{p}(u, v)$, or we might even say that the parametric representation $\boldsymbol{p} = \boldsymbol{p}(u, v)$ is contained in $S$ when the image of $\boldsymbol{p} = \boldsymbol{p}(u, v)$ is a subset of $S$.

### B. Fundamental Forms

A surface in 3-D space is uniquely determined by certain local invariant quantities known as the first and second fundamental forms [8], [10], [13]. Let $\boldsymbol{p} = \boldsymbol{p}(u, v)$ be a parametric surface patch of class $\geq 1$; then, the first fundamental form is

$$\begin{aligned} I &= d\boldsymbol{p} \cdot d\boldsymbol{p} \\ &= (\boldsymbol{p}_u du + \boldsymbol{p}_v dv) \cdot (\boldsymbol{p}_u du + \boldsymbol{p}_v dv) \\ &= (\boldsymbol{p}_u \cdot \boldsymbol{p}_u) du^2 + 2(\boldsymbol{p}_u \cdot \boldsymbol{p}_v) du dv + (\boldsymbol{p}_v \cdot \boldsymbol{p}_v) dv^2 \\ &= E du^2 + 2F du dv + G dv^2 \end{aligned} \tag{2.4}$$

where

$$E = \boldsymbol{p}_u \cdot \boldsymbol{p}_u, F = \boldsymbol{p}_u \cdot \boldsymbol{p}_v, G = \boldsymbol{p}_v \cdot \boldsymbol{p}_v. \tag{2.5}$$

$E$, $F$, and $G$ are known as the coefficients of the first fundamental form. Suppose $\boldsymbol{p} = \boldsymbol{p}(u, v)$ is a surface patch of class $\geq 2$. The unit normal to a surface at a point $\boldsymbol{p}(u, v)$ is $\boldsymbol{n}(u, v) = \frac{\boldsymbol{p}_u \times \boldsymbol{p}_v}{|\boldsymbol{p}_u \, times \, \boldsymbol{p}_v|}$, which is a function with differential $d\boldsymbol{n} = \boldsymbol{n}_u du + \boldsymbol{n}_v dv$. Note that $d\boldsymbol{n}$ is a vector parallel to the tangent plane at $\boldsymbol{p}(u, v)$. This follows from $0 = d(1) = d(\boldsymbol{n} \cdot \boldsymbol{n}) = 2d\boldsymbol{n} \cdot \boldsymbol{n}$. The second fundamental form is

$$\begin{aligned} II &= -d\boldsymbol{p} \cdot d\boldsymbol{n} \\ &= -(\boldsymbol{p}_u du + \boldsymbol{p}_v dv)(\boldsymbol{n}_u du + \boldsymbol{n}_v dv) \\ &= -\boldsymbol{p}_u \cdot \boldsymbol{n}_u du^2 - (\boldsymbol{p}_u \cdot \boldsymbol{n}_v + \boldsymbol{p}_v \cdot \boldsymbol{n}_u) du dv - \boldsymbol{p}_v \cdot \boldsymbol{n}_v dv^2 \\ &= L du^2 + 2M du dv + N dv^2 \end{aligned} \tag{2.6}$$

where

$$L = -\boldsymbol{p}_u \cdot \boldsymbol{n}_u, \quad M = -\frac{1}{2}(\boldsymbol{p}_u \cdot \boldsymbol{n}_v + \boldsymbol{p}_v \cdot \boldsymbol{n}_u), \quad N = -\boldsymbol{p}_v \cdot \boldsymbol{n}_v. \tag{2.7}$$

### C. Curvatures

Let $P$ be a point on a surface $\boldsymbol{p} = \boldsymbol{p}(u, v)$ of class $\geq 2$, and let $\boldsymbol{p} = \boldsymbol{p}(u(t), v(t))$ be a curve $C$ that lies on the surface and passes through $P$. The normal curvature vector to $C$ at $P$ is the vector projection of the curvature vector $\boldsymbol{k}$ of $C$ at $P$ onto the normal $\boldsymbol{n}$ at $P$, i.e.

$$\boldsymbol{k}_n = (\boldsymbol{k} \cdot \boldsymbol{n})\boldsymbol{n}. \tag{2.8}$$

Notice that $\boldsymbol{k}_n$ is independent of the sense of $\boldsymbol{n}$ or of $C$. The component of $\boldsymbol{k}_n$ in the direction of $\boldsymbol{n}$ is called the *normal curvature*

of $C$ at $P$, that is

$$k_n = \boldsymbol{k} \cdot \boldsymbol{n}. \tag{2.9}$$

Here, the sign of $k_n$ depends on the sense of $\boldsymbol{n}$, but it is independent of the sense of $C$.

Note that the unit tangent to $C$ at $P$ is $\boldsymbol{t} = \frac{d\boldsymbol{p}}{ds} = \frac{d\boldsymbol{p}/dt}{|d\boldsymbol{p}/dt|}$, and the curvature vector is $\boldsymbol{k} = \frac{d\boldsymbol{t}}{ds} = \frac{d\boldsymbol{t}/dt}{|d\boldsymbol{p}/dt|}$. Thus, $0 = \frac{d}{dt}(\boldsymbol{t} \cdot \boldsymbol{n}) = \frac{d\boldsymbol{t}}{dt} \cdot \boldsymbol{n} + t \cdot \frac{d\boldsymbol{n}}{dt}$ since $\boldsymbol{t}$ is perpendicular to $\boldsymbol{n}$ along the curve.

It follows that

$$k_n = \frac{L(du/dt)^2 + 2M(du/dt)(dv/dt) + N(dv/dt)^2}{E(dv/dt)^2 + 2F(du/dt)(dv/dt) + G(dv/dt)^2}. \tag{2.10}$$

Note that $k_n$ depends only on $\frac{du/dt}{dv/dt}$, which is the direction of the tangent line to $C$ at $P$. Otherwise, $k_n$ is a function of the fundamental forms $I$ and $II$, which depend only on $P$.

All curves through a point $P$ on a surface tangent to the same line through $P$ have the same normal curvature at $P$ [8]. Since the normal curvature to $C$ at $P$ depends only on $P$ and the direction of the tangent line to $C$ at $P$, and we can say that the normal curvature in the direction $du : dv$, $du$, and $dv$ are not both zero, we have

$$k_n = \frac{Ldu^2 + 2Mdudv + Ndv^2}{Edu^2 + 2Fdudv + Gdv^2}. \tag{2.11}$$

Notice that the above form is simply $k_n = \frac{II}{I}$. $du : dv$ are the direction numbers of the line in the tangent plane parallel to $\boldsymbol{p}_u du + \boldsymbol{p}_v dv$. $du : dv$ and $du' : dv'$ determine the same line if and only if they are proportional. It is clear that $k_n$ is invariant in the same sense as $I$ and $II$. $k_n$ does not change sign under a parametric transformation that preserves the sense of $\boldsymbol{n}$, and $ksubn$ changes sign under a parametric transformation, which reverses the sense of $\boldsymbol{n}$.

Since $I \geq 0$, $k_n$ is positive, negative, or zero together with $II$, that is, if $P$ is elliptic, then $k_n \neq 0$ and remains the same sign for all $du : dv$ at $P$. If $P$ is a hyperbolic point, then $k_n$ is positive, negative, or zero, depending on $du : dv$. If $P$ is a parabolic point, $k_n$ remains the same sign and is zero for the direction for which $II = 0$. If $P$ is planar, $k_n = 0$ in all directions.

The two perpendicular directions for which the values of $k_n$ take on maximum and minimum values are called the principal directions. The maximum and minimum values of normal curvatures $k_1$ and $k_2$ are called the *principal curvatures* [10]. A point on the surface at which $k_n$ is constant is called an umbilical point. If $k_n = constant \neq 0$, it is called an elliptic umbilical point. If $k_n = constant = 0$, it is called a parabolic umbilical point. Therefore, if all points of a connected surface $S$ are umbilical, then $S$ is either contained in a sphere or in a plane [13]. The principal curvatures are roots of

$$(EG - F^2)k^2 - (EN + GL - 2FM)k + (LN - M^2) = 0. \tag{2.12}$$

The average of the roots of the above equation is the mean curvature, and the product is the Gaussian curvature.

## III. Data Structures for Our System

In our vision task, the knowledge the system needs is the descriptions about objects. We already have the B-rep about objects; however, it is suitable for producing images from object models but not suitable for vision tasks.

A model database is a set $M = m_1, m_2, \ldots, m_n$, where each $m_i$, $1 \leq i \leq n$ is the B-rep for a solid object.

A B-rep is a set $m_i = f_{i1}, f_{i2}, \ldots, f_{ip_i}$, where $f_{ij}$, $1 \leq j \leq p_i$ is a surface patch. This set of surface patches cover the boundary of $m_i$.

Each model is described by a set of surface patches and the implicit spatial relationships among the surface patches. This information is sufficient for generating an image from the models, which will be used in our verification process during object recognition and localization. We use a data structure known as a curvature map to represent the surface shape. A curvature map can be viewed as a mesh on a surface patch as shown in Fig. 1. We compute the Gaussian curvature $K$ and the magnitude of mean curvature $|H|$ at each node of the mesh in Fig. 1. In the system, a curvature map is, in fact, a three-layer structure as shown in Fig. 2. One layer stores the spatial coordinates of each node of the mesh. The other two layers are for the Gaussian curvature and the magnitude of mean curvature at the corresponding points. Each curvature map is reorganized into the structure shown in Fig. 3. The Gaussian curvature layer and the magnitude of mean curvature layer are sorted arrays. Each value in either curvature layer is associated with a point coordinate. Note that $K_i$ and $|H_i|$ are not necessarily associated with the same point.

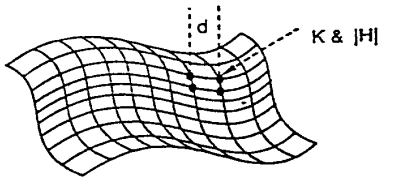


Fig. 1. Curvature map as a mesh on a surface patch.

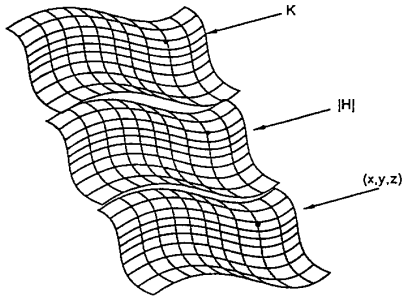


Fig. 2 Curvature map as a three-layer structure.

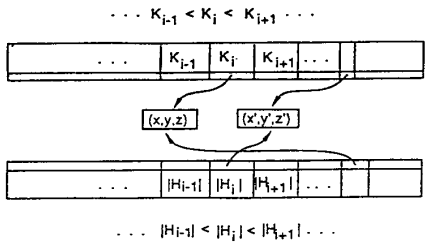


Fig. 3. Physical structure of curvature map.

To perform recognition reasoning, each plane is associated with its surface normal. The relationship between plane normals within a model can help recognition reasoning.

## IV. Control Strategy

The vision process in an autonomous robot normally involves the recognition of complex objects in scenes and the problems of estimating the position and orientation of an object. The above process is computationally intensive, and new models are needed that permit efficient ways of recognizing and localizing 3-D objects.

### A. Recognition and Localization

Suppose there are $n$ objects $O = \{o_1, o_2, \ldots, o_n\}$ in a scene, and there are $k$ models $M = \{m_1, m_2, \ldots, m_k\}$ in the model database. Our task is to find a matching $o_i = TRANSFORM(m_j)$, where $o_i \in O$, $m_j \in M$, and $TRANSFORM$ is a transformation, that is, we should find a matching between $o_i$ and $m_j$ for the recognition task and find the transformation $TRANSFORM$ for the localization task. Since we can only find a set of surface segments $S = \{s_1, s_2, \ldots, s_t\}$ in an image, the task becomes one of finding the membership of an image surface segment in the set of models, i.e., finding $s_i \in TRANSFORM(m_j)$, where $s_i \in S$, and $m_j \in M$. Since each model is described in terms of a set of surface patches $m_i = f_{i1}, f_{i2}, \ldots, f_{ip_i}$, the task becomes one of finding the membership of an image surface segment in a subset of the surface patches of a certain model, i.e., $s_i \in TRANSFORM(m_j')$, where $s_i \in S$, and $m_j' \subseteq m$. Since it is not cost effective to search all possible combinations of image surface segments with model surface patches, the task becomes an AI search problem. Since heuristic search is well defined and understandable, we introduce our approach in terms of heuristic search.

### B. Main Algorithm

Our algorithm is a heuristic search procedure using an evaluation function to direct search through the state space. A curved surface in an image contains more heuristic information by itself than a planar surface, which can be understood intuitively. We focus our proposed technique on curved surface segments differently from planar surface segments in an image. First, we use information from curved surface segments to direct the search. If no curved segments can be used, the planar surface segments are used to continue searching. The main algorithm is as follows.

**MAIN ALGORITHM**

1. If the input image is empty, then exit.
2. Let $s$ be the set of all models, where each consists of a set of surface patches. Set $TEMP$ and $P$ to $\phi$.
3.* Find a curved surface segment $p'$ in the input image such that $p'$ is close to one segment in $P$, and $p'$ is not in $TEMP$. Add $p'$ to $P$. If not found, go to step 13.
4.* Obtain the subset $s' \subseteq s$ such that $p' \in P$ is matched to the surface patches of $m_i$ for all $m_i \in s'$.
5. If $|s'| = 0$, delete $p'$ from $P$, and place $p'$ in $TEMP$. (If $|P| = 0$, erase $p'$ from the image.) Go to step 3.
6.* If $|s'| = 1$, a hypothesis is formed. Go to step 9 for verification.
7.* If there are three surface segments in $P$ that match with the same model in $s'$, then find the model $m$ in $s'$ that matches the most segments in $P$. Delete $m$ from $s'$. Go to step 9 for verification.
8. Let $s = s'$. Go to step 3.
9.* If verification succeeds, erase $TRANSFORM(m)$ from the input image. Go to step 1.
10.* Check whether surfaces of $m$ can match to segments in $P$ at different positions. If another matching position is found, go to step 9.
11. If $|s'| = 0$, fail to recognize all segments in $P$. Erase all $p' \in P$ from the image. Go to step 1.
12. Go to step 6.
13.* If $|P| > 0$, find a planar surface segment in the image close to one segment in $P$. If found, go to step 4; otherwise, go to step 15.

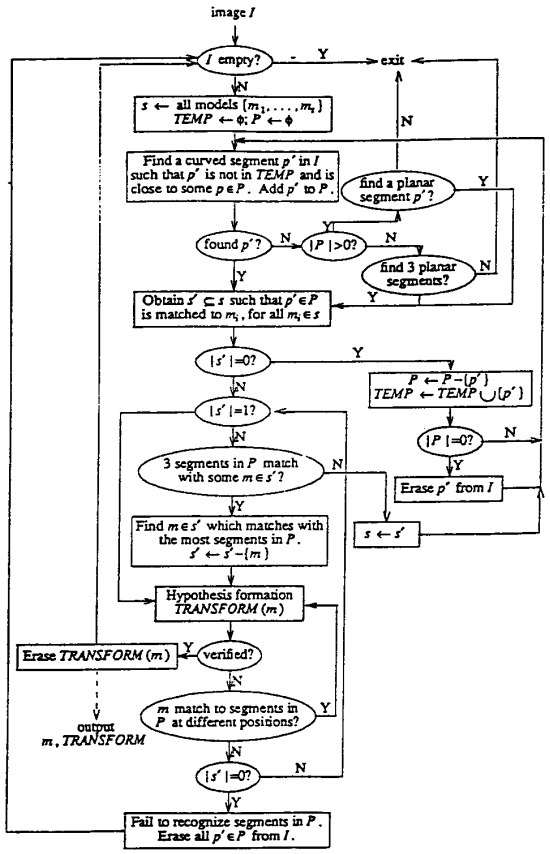


Fig. 4. Flow diagram of the main algorithm.

14.* Find three planar surface segments close to each other. If found, go to step 4.
15. Exit with failure.

Note: Steps with * will be discussed in detail in the following sections.

There are many tradeoffs in application systems. Heuristics help reduce the amount of search. However, applying heuristics to find the search directions is also expensive. If we use more heuristics, the search is more efficient. However, if we use more heuristics, the cost of the evaluation function will be higher. It is very difficult to design an evaluation function that not only performs efficiently but also contains a great deal of heuristic information. Hence, the total cost of a search system can be classified as two classes: one is the cost of the search process, and another is the cost of the heuristic evaluation operator. The total cost of a search system can be shown informally in Fig. 4. In designing a search system, we first have to find tradeoffs, considering the application requirements and the factors of the search system. One way to design heuristic evaluation functions is to design multiple functions to deal with different situations so that each function is efficient. Our approach is to use different heuristic information and different evaluation functions to deal with curved surface segments and planar surface segments in the input image, which has been proven to be efficient.

### C. Image Segmentation

This section discusses techniques used to obtain a surface segment in steps 3, 13, and 14. The knowledge about model surface patches

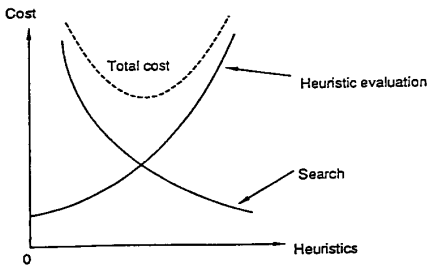


Fig. 5. Search system cost.

is represented as a Gaussian curvature map and a magnitude of mean curvature map for curved surfaces as well as angles between surface normals for planar surface patches. Therefore, the useful features in the image are the curvatures of curved surface segments and angles between planar surface segments. In order to find these features, the simplest method is to use an approximation function to apply to a small window of pixels to find the curvatures and then classify a region of neighboring pixels into surface segments such that each segment contains a smooth surface. Suppose that we use a $5 \times 5$ window to obtain the curvatures of the central pixel of the window as shown in Fig. 5. Let us use an explicit function $z = f(x, y)$ to approximate the window in order to obtain the first and second derivatives on the central pixel to compute the curvatures. Let

$$z = a_0x^3 + a_1x^2y + a_2xy^2 + a_3y^3 + a_4x^2 + a_5xy + a_6y^2 + a_7x + a_8y + a_9. \tag{5.1}$$

Let the center of the window be $(x, y) = (0, 0)$, and every pixel is one unit, as shown in Fig. 5. We can use a least-squares approach to find the approximation efficiently. Let

$$\boldsymbol{x} = [a_0, a_1, \ldots, a_9]^T$$

$$\boldsymbol{y} = [z(-2,-2), z(-1,-2), \ldots, z(2,-2), z(-2,-1), z(-1,-1), \ldots, z(2,-1), \ldots, z(2,2)]^T$$

$$A = \begin{bmatrix} x_0^3 & x_0^2y_0 & x_0y_0^2 & y_0^3 & \ldots & x_0 & y_0 & 1 \\ x_1^3 & x_1^2y_1 & x_1y_1^2 & y_1^3 & \ldots & x_1 & y_1 & 1 \\ . & . & . & . & \ldots & . & . & . \\ . & . & . & . & \ldots & . & . & . \\ . & . & . & . & \ldots & . & . & . \\ x_{24}^3 & x_{24}^2y_24 & x_{24}y_{24}^2 & y_{24}^3 & \ldots x_{24} & y_{24} & 1 & \end{bmatrix}$$

where $(x_0, y_0) = (-2, -2)$, $(x_1, y_1) = (-1, -2)$, $(x_2, y_2) = (0, -2), \ldots, (x_{24}, y_{24}) = (2, 2)$. We have an overdetermined system

$$A\boldsymbol{x} = \boldsymbol{y}. \tag{5.2}$$

The least-squares method is to find an $\boldsymbol{x}$ such that $|A\boldsymbol{x} - \boldsymbol{y}|^2 =$ minimum

Multiply $A^T$ to both sides of (5.4); thus, we have

$$A^TA\boldsymbol{x} = A^T\boldsymbol{y} \tag{5.3}$$

where

$$A^TA =$$

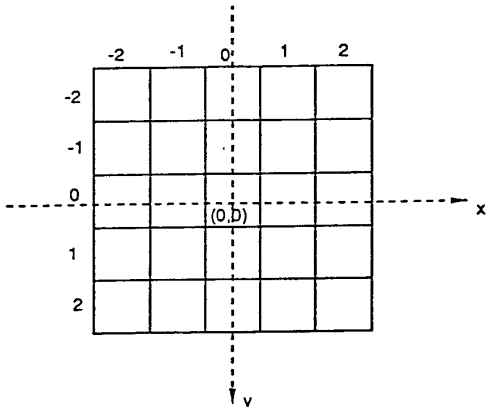


Fig. 6. Window for approximation to obtain curvatures.

$$\begin{bmatrix} 650 & 0 & 340 & 0 & 0 & 0 & 0 & 170 & 0 & 0 \\ 0 & 340 & 0 & 340 & 0 & 0 & 0 & 0 & 100 & 0 \\ 340 & 0 & 340 & 0 & 0 & 0 & 0 & 100 & 0 & 0 \\ 0 & 340 & 0 & 650 & 0 & 0 & 0 & 0 & 170 & 0 \\ 0 & 0 & 0 & 0 & 170 & 0 & 100 & 0 & 0 & 50 \\ 0 & 0 & 0 & 0 & 0 & 100 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 100 & 0 & 170 & 0 & 0 & 50 \\ 170 & 0 & 100 & 0 & 0 & 0 & 0 & 50 & 0 & 0 \\ 0 & 100 & 0 & 170 & 0 & 0 & 0 & 0 & 50 & 0 \\ 0 & 0 & 0 & 0 & 50 & 0 & 50 & 0 & 0 & 25 \end{bmatrix}.$$

Since $A^TA$ is nonsingular, we multiply both sides of (5.1) by $(A^TA)^{-1}$.

$$\boldsymbol{x} = (A^TA)^{-1}A^T\boldsymbol{y}. \tag{5.4}$$

Let $B = (A^TA)^{-1}A^T$. Then

$$\boldsymbol{x} = B\boldsymbol{y} \tag{5.5}$$

where $B$ is a $10 \times 25$ matrix computed before the system starts. $\boldsymbol{y}$ is the data from the window. Therefore, the coefficients of the approximation function (5.3) are easily obtained. Since the purpose of finding the approximation function is to compute the curvatures at the central pixel of the window, we can derive a more efficient way to perform the task. The first derivatives of (5.3) are

$$z_x = 3a_0x^2 + 2a_1xy + a_2y^2 + 2a_4x + a_5y + a_7$$

$$z_y = a_1x^2 + 2a_2xy + 3a_3y^2 + a_5x + 2a_6y + a_8$$

and the second derivatives of (5.3) are

$$z_{xx} = 6a_0x + 2a_1y + 2a_4$$

$$z_{yy} = 2a_2x + 6a_3y + 2a_6$$

$$z_{xy} = 2a_1x + 2a_2y + a_5 = z_{yx}.$$

Substituting the coordinates of the center of the window that are $(0, 0)$, at the central pixel of the window, we have $z_x = a_7, z_y = a_8, z_{xx} = 2a_4, z_{yy} = 2a_6, z_{xy} = a_5$.

If we represent $B$ in (5.7) as the column vector $[B_0, B_1, B_2, \ldots B_9]$, where $B_i$, $0 \leq i \leq 9$, is a $1 \times 25$ matrix that is the $i$th row of $B$, then we have $z_x = a_7 = B_7\boldsymbol{y}$, $z_y = a_8 = B_8\boldsymbol{y}$, $z_{xx} = 2a_4 = 2B_4\boldsymbol{y}$, $z_{yy} = 2a_6 = 2B_6\boldsymbol{y}$, and $z_xy = a_5 = B_5\boldsymbol{y}$.

Since $\boldsymbol{y}$ is a $5 \times 5$ window on the image, we can represent the above as $5 \times 5$ window operators so that we just apply the operators to the window. An operator is represented as a $5 \times 5$ matrix. When applying the operator, we simply multiply the corresponding elements

in the window and then add them together. The operators $z_x, z_y, \ldots$ can now be calculated and used to obtain the curvature. In this way, we can obtain curvatures for each pixel in a small region. A smooth surface is a surface that has continuous first and second derivatives. Therefore, we should not include edge points in the surface segment region in order to obtain a smooth surface segment. Since edge points must have high curvature, we can determine whether a point is on an edge by testing the principal curvatures at that point. From the roots of (2.12), we have $k_1 = H - (H^2 - K)^{\frac{1}{2}}$ $k_2 = H + (H^2 - K)^{\frac{1}{2}}$.

If either $|k_1|$ or $|k_2|$ is greater than a threshold $T > 0$, the point is on an edge. The selection of $T$ is by experimentation. Finally, we obtain a local curvature map of a smooth surface segment in the image. The surface nature can be determined according to the curvatures. If the whole map has $|K| \leq K_{zero}$ and $|H| \leq H_{zero}$, the surface is a plane. $K_{zero}$ and $H_{zero}$ are also thresholds for testing the zero of $K$ and $H$ since $K$ and $H$ cannot really be zero in the input image due to noise and quantization effects. Since $K$ is the product of principal curvatures and $H$ is the average of principal curvatures, we let $K_{zero} \geq H^2_{zero}$. These thresholds are also selected through experimentation. After the surface nature and the local curvature map are obtained, they are used as heuristic information in the evaluation function at step 4 in the main algorithm.

### D. Surface Matching Evaluation

In this section, we discuss how to perform steps 4 and 10 of the main algorithm.

After the curvature map of a surface segment is obtained, we can evaluate how close the curvature map matches with the curvature maps of model surface patches in set $s$. $s$ has all models initially and then contains a subset of models after one or more iterations. If the curvature map of the surface segment indicates the surface segment as a plane, then the surface segment can match to all planar surface patches of models in $s$. Therefore, at step 4, all models not including any planar surface patches are deleted from $s$ to form $s'$. If the surface segment is curved, we need more work to select the subset $s'$ of $s$ at step 4. The surface segment curvature map can be viewed as a three-layer data structure. The first layer is the surface segment from the image where every pixel is associated with its coordinates $(x, y, z)$ in the 3-D space, i.e., the pixel coordinates in the image associates with $(x, y)$, and the pixel value is $z$, as shown in Fig. 6. The other two layers are the Gaussian curvatures and the magnitudes of mean curvature, which are the same size as the surface segment layer. Every pixel in the segment has its Gaussian curvature and the magnitude of its mean curvature in these two layers as shown in Figs. 7 and 8, respectively. Now let us take a model $m$ from $s$ to check whether the surface segment can match to its surface patches. Since the surface segment is smooth, the center shape of the surface segment is least affected by noise. We first locate the possible positions of the center pixel of the surface segment on the model $m$. From the initial positions, if we can develop a local fitting of the surface segment to the model surfaces, it is possible that the surface segment matches with the model. Then, the fitting position of the center of the segment is also recorded. In the previous section, we discussed the knowledge representation of surface shapes of the models. The surface shape is represented as a curvature map for a surface patch. The curvature maps are ordered lists that are sorted by values of $K$ and $|H|$. The initial possible positions of the segment center are located in the following way. Suppose that the center of the surface segment is on the $(0, 0)$ pixel of the segment curvature map, as shown in Figs. 7 and 8. Let

$$\alpha = \max\left\{|K_{(0,0)} - K_{(i,j)}| : i, j = -1, 0, 1\right\}$$
$$\beta = \max\left\{||H|_{(0,0)} - |H|_(i, j)| : i, j = -1, 0, 1\right\}.$$

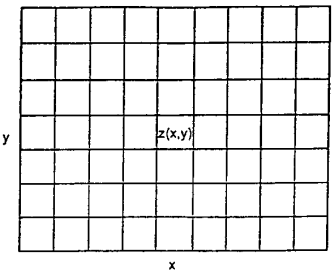


Fig. 7. Surface segment layer.

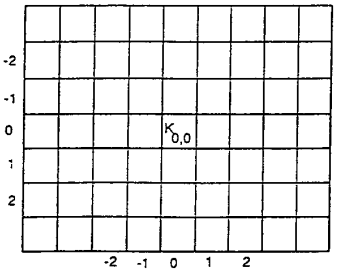


Fig. 8. Gaussian curvature layer.

Thus, $\alpha$ is the maximum difference between the central $K$ value and its neighboring $K$ value, and $\beta$ is the maximum difference between the central $|H|$ value and its neighboring values.

A point $\boldsymbol{p}$ on the model with Gaussian curvature $K_{\boldsymbol{p}}$ and magnitude of mean curvature $|H|_{\boldsymbol{p}}$ is a possible matching position of the center of the surface segment if $|K_{(0,0)} - K_{\boldsymbol{p}}| < \alpha$ and $||H|_{(0,0)} - |H|_{\boldsymbol{p}}| < \beta$. These points can be found efficiently by binary search from model surface curvature maps represented as ordered lists. Let these initial points be in set $SP$, let the distance between two neighboring pixels be $\frac{1}{2}\epsilon$, and let the noise rate be $\sigma$. Then, we use the following procedure to check whether the points in $SP$ can be selected as corresponding points to the center of the surface segment.

**Procedure Fitting**

1. $n$ is the total number of pixels of the surface segment. $u = 0$, which is the number of pixels unmatched.
2. If $SP = \phi$, exit with failure.
3. Take $\boldsymbol{p}$ from $SP$. $SP = SP - \boldsymbol{p}$.
4. For every noncentral position $(i, j)$ of the surface segment curvature map
5. $\alpha' = \max\left\{|K_{(i,j)} - K_{(i+i',j+j')}| : i', j' = -1, 0, 1\right\}$
6. $\beta' = \max\left\{||H|_{(i,j)} - |H|_{(i+i',j+j')}| : i', j' = -1, 0, 1\right\}$.
7. For every point $\boldsymbol{p}'$ on the model $m$ with
8. $|K_{(i,j)} - K_{\boldsymbol{p}'}| < \alpha'$ and $||H|_{(i,j)} - |H|_{\boldsymbol{p}'}| < \beta'$.
9. If $|\text{distance}(seg\_p_{(0,0)}, seg\_p_{(i,j)}) - \text{distance}(\boldsymbol{p}', \boldsymbol{p})| < \epsilon$, go to step 4.
10. End_for /* step 7 */.
11. $u = u + 1$.
12. If $\frac{u}{n} > \sigma$, then go to step 2.
13. End_for /* step 4 */.
14. Exit with success.

Note that the procedure **Fitting** only finds a very possible match when it exits successfully. The procedure only does checking on the distance between points, which is a necessary condition for a match but not a sufficient condition. If we want to inspect all conditions to guarantee the exact match, however, the process will be very expensive. Referring to Fig. 4, we have a tradeoff between the evaluation cost and the search cost.

### E. Hypothesis Formation

In this section, we discuss the techniques in steps 6 and 7 of the main algorithm, which are used to form a hypothesis for verification in step 9. We defined our task as finding matchings between surface segments in the image and a transformation of model surfaces $s_i \in TRANSFORM(m_j)$, where $s_i \in s$, which is a set of surface segments, and $m_j \in M$, which is a set of models. We have discussed how to find a possible membership of $s_i$ on $m_j$. However, in order to verify the matching, we have to find $TRANSFORM$ and then check whether $s_i$ is really on $TRANSFORM(m_j)$. Thus, at this point, we simply need to find $TRANSFORM$ to obtain the hypothesis.

If $|s'| > 1$ at step 7, we select a model $m$ from $s'$ such that $m$ has the most possible matching surface segments in $P$ to form the hypothesis. If there are three curved surface segments matched to $m$, and their possible positions on $m$ have been found in previous steps as discussed in Section V-IV, then we perform a distance checking between these points similar to that in procedure **Fitting**. If the distance checking fails, the hypothesis cannot be formed. Otherwise, we can obtain $TRANSFORM$ from the relationship of the central points of the three surface segments and the corresponding positions on the model $m$ since three noncollinear points uniquely determine a 3-D spatial transformation. If there is only one curved surface segment matched with model $m$, which is possible at step 6, then we use the central point of the surface segment and two additional noncentral points to obtain the transformation. Let the three points from the surface segments be $(sx_0, sy_0, sz_0)$, $(sx_1, sy_1, sz_1)$, and $(sx_2, sy_2, sz_2)$. Let their corresponding points on the model $m$ be $(x_0, y_0, z_0)$, $(x_1, y_1, z_1)$, and $(x_2, y_2, z_2)$.

A transformation of points in 3-D space is represented explicitly as

$$[x'y'z'] = [xyz]R + T \tag{5.6}$$

where $R$ is a $3 \times 3$ rotation matrix, and $T$ is a $1 \times 3$ translation matrix. The rotation matrices corresponding to the $z$, $x$, and $y$ axes are, respectively

$$R_z(\theta) = \begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix};$$

$$R_x(\theta) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta & \sin\theta \\ 0 & -\sin\theta & \cos\theta \end{bmatrix};$$

$$R_y(\theta) = \begin{bmatrix} \cos\theta & 0 & -\sin\theta \\ 0 & 1 & 0 \\ \sin\theta & 0 & \cos\theta \end{bmatrix}.$$

Thus, $R$ in (5.10) is the composition of an arbitrary sequence of rotations about the $x, y,$, and $z$ axes. We have

$$R = \begin{bmatrix} r_{00} & r_{01} & r_{02} \\ r_{10} & r_{11} & r_{12} \\ r_{20} & r_{21} & r_{22} \end{bmatrix} \quad T = [t_x t_y t_z].$$

To find the transformation of the three points, we can easily pin down the translation: $T = [t_x t_y t_z] = [sx_0 - x_0 \; sy_0 - y_0 \; sz_0 - z_0]$.

When we substitute the three points in (5.10), we have

$$[sx_0 sy_0 sz_0] = [x_0 y_0 z_0]R + T \tag{5.7}$$

$$[sx_1 sy_1 sz_1] = [x_1 y_1 z_1]R + T \tag{5.8}$$

$$[sx_2 sy_2 sz_2] = [x_2 y_2 z_2]R + T \tag{5.9}$$

From (5.7)–(5.9), we get

$$\begin{aligned} sx_0 &= x_0 r_{00} + y_0 r_{10} + z_0 r_{20} + t_x, \\ sy_0 &= x_0 r_{01} + y_0 r_{11} + z_0 r_{21} + t_y, \\ sz_0 &= x_0 r_{02} + y_0 r_{12} + z_0 r_{22} + t_z. \\ sx_1 &= x_1 r_{00} + y_1 r_{10} + z_1 r_{20} + t_x, \\ sy_1 &= x_1 r_{01} + y_1 r_{11} + z_1 r_{21} + t_y, \\ sz_1 &= x_1 r_{02} + y_1 r_{12} + z_1 r_{22} + t_z. \\ sx_2 &= x_2 r_{00} + y_2 r_{10} + z_2 r_{20} + t_x, \\ sy_2 &= x_2 r_{01} + y_2 r_{11} + z_2 r_{21} + t_y, \\ sz_0 &= x_2 r_{02} + y_2 r_{12} + z_2 r_{22} + t_z. \end{aligned}$$

From the above three linear systems, we obtain $R$. Therefore, the hypothesis is formed, which is a matching between surfaces and their transformation. If there are only planar segments matched with model $m$, we need three nonparallel planes to determine a transformation in 3-D space. To obtain a plane equation

$$Ax + By + Cz + D = 0 \tag{5.10}$$

we need three noncollinear points on the plane. Given three noncollinear points $(x_1, y_1, z_1)$, $(x_2, y_2, z_2)$, and $(x_3, y_3, z_3)$ on the plane, we have

$$\begin{aligned} Ax_1 + By_1 + Cz_1 + D &= 0, \\ Ax_2 + By_2 + Cz_2 + D &= 0, \\ Ax_3 + By_3 + Cz_3 + D &= 0. \end{aligned}$$

Since the three points are not collinear, the equations can be solved for $A, B, C$, and $D$ by arbitrarily assigning a value to one of them and then solving the resulting three equations in three unknowns. A better way to obtain (5.14) is to select any point $(x, y, z)$ on the plane; then, we have

$$\begin{aligned} Ax + By + Cz + D &= 0, \\ Ax_1 + By_1 + Cz_1 + D &= 0, \\ Ax_2 + By_2 + Cz_2 + D &= 0, \\ Ax_3 + By_3 + Cz_3 + D &= 0. \end{aligned}$$

If there is a nontrivial (nonzero) solution to this homogeneous system, the determinant of its coefficients must be zero. Expanding by cofactors about the first row, we have

$$A'x + B'y + C'z + D' = 0 \tag{5.11}$$

where

$$A' = \det \begin{pmatrix} y_1 & z_1 & 1 \\ y_2 & z_2 & 1 \\ y_3 & z_3 & 1 \end{pmatrix};$$

$$B' = -\det \begin{pmatrix} x_1 & z_1 & 1 \\ x_2 & z_2 & 1 \\ x_3 & z_3 & 1 \end{pmatrix};$$

$$C' = \det \begin{pmatrix} x_1 & y_1 & 1 \\ x_2 & y_2 & 1 \\ x_3 & y_3 & 1 \end{pmatrix};$$

$$D' = -\det \begin{pmatrix} x_1 & y_1 & z_1 \\ x_2 & y_2 & z_2 \\ x_3 & y_3 & z_3 \end{pmatrix}.$$

Thus, (5.11) is the equation we want. The three points cannot be collinear. If all cofactors in (5.11) are zero, collinearity occurs.

The surface normal of (5.10) is

$$\boldsymbol{n} = \frac{(A, B, C)}{(A^2 + B^2 + C^2)^{1/2}}.$$

Let the three plane segments be

$$A'_1 x + B'_1 + C'_1 z + D'_1 = 0 \quad (5.12)$$
$$A'_2 x + B'_2 + C'_2 z + D'_2 = 0 \quad (5.13)$$
$$A'_3 x + B'_3 + C'_3 z + D'_3 = 0 \quad (5.14)$$

and their surface normals be

$$\boldsymbol{n}'_1 = \frac{(A'_1, B'_1, C'_1)}{(A'^2_1 + B'^2_1 + C'^2_1)^{1/2}},$$
$$\boldsymbol{n}'_2 = \frac{(A'_2, B'_2, C'_2)}{(A'^2_2 + B'z^2_2 + C'^2_2)^{1/2}},.$$
$$\boldsymbol{n}'_3 = \frac{(A'_3, B'_3, C'_3)}{(A'^2_3 + B'^2_3 + C'^2_3)^{1/2}}$$

In order to determine a 3-D transformation uniquely, we must have $\boldsymbol{n}_i \neq \boldsymbol{n}_j$, $1 \leq i, j \leq 3$, $i \neq j$.

The angle between two planes is obtained from $\cos\theta = \boldsymbol{n}_i \cdot \boldsymbol{n}_j =$

$$\frac{A_i A_j + B_i B_j + C_i C_j}{(A_i^2 + B_i^2 + C_i^2)^{1/2}(A_j^2 + B_j^2 + C_j^2)^{1/2}}.$$

Since we have had all the angle information between planes in a model, we should inspect the angles between any two planar surface segments and compare them with the angles between planes in the model. If a matching can be found, we can proceed to find the transformation from the corresponding planes to form a hypothesis. Suppose the three corresponding planes on the model $m$ are

$$A_1 x + B_1 + C_1 z + D_1 = 0 \quad (5.15)$$
$$A_2 x + B_2 + C_2 z + D_2 = 0 \quad (5.16)$$
$$A_3 x + B_3 + C_3 z + D_3 = 0. \quad (5.17)$$

Let us first determine the rotation matrix of the transformation, i.e., to rotate the model $m$ such that after rotation (5.15) is parallel to (5.12), (5.16) is parallel to (5.13), and (5.17) is parallel to (5.14). We use the rotation matrix in (5.10). Substitute the transformation in (5.15). We have

$$A_1(xr_{00} + yr_{10} + zr_{20} + t_x) - B_1(xr_{01} + yr_{11} + zr_{21} + t_y) + C_1(xr_{02} + yr_{12} + zr_{22} + t_z) + D_1 = 0.$$

The new equation is

$$(A_1 r_{00} + B_1 r_{01} + C_1 r_{02})x + (A_1 r_{10} + B_1 r_{11} + C_1 r_{12})y + (A_1 r_{20} + B_1 r_{21} + C_1 r_{22})z + D_1 + t_x + t_y + t_z = 0. \quad (5.18)$$

Since (5.18) is parallel to (5.12), we have

$$\frac{A_1 r_{00} + B_1 r_{01} + C_1 r_{02}}{A'_1} = \frac{A_1 r_{10} + B_1 r_{11} + C_1 r_{12}}{B'_1} = \frac{A_1 r_{20} + B_1 r_{21} + C_1 r_{22}}{C'_1}. \quad (5.19)$$

Similarly, from (5.16) and (5.17), we have

$$\frac{A_2 r_{00} + B_2 r_{01} + C_2 r_{02}}{A'_2} = \frac{A_2 r_{10} + B_2 r_{11} + C_2 r_{12}}{B'_2} = \frac{A_2 r_{20} + B_2 r_{21} + C_2 r_{22}}{C'_2} \quad (5.20)$$

and

$$\frac{A_3 r_{00} + B_3 r_{01} + C_3 r_{02}}{A'_3} = \frac{A_3 r_{10} + B_3 r_{11} + C_3 r_{12}}{B'_3} = \frac{A_3 r_{20} + B_3 r_{21} + C_3 r_{22}}{C'_3}. \quad (5.21)$$

Thus, we can solve nine unknowns in nine linear equations of (5.19)–(5.21). In order to let (5.18) and (5.12) represent the same plane, we have other conditions shown as follows:

$$\frac{D_1 + t_x + t_y + t_z}{D'_1} = \frac{A_1 r_{00} + B_1 r_{01} + C_1 r_{02}}{A'_1}$$
$$\frac{D_2 + t_x + t_y + t_z}{D'_2} = \frac{A_2 r_{00} + B_2 r_{01} + C_2 r_{02}}{A'_2}$$
$$\frac{D_3 + t_x + t_y + t_z}{D'_3} = \frac{A_3 r_{00} + B_3 r_{01} + C_3 r_{02}}{A'_3}$$

which determine the translation matrix $T = [t_x t_y t_z]$.

Therefore, we obtain the transformation from the corresponding matching surface to a hypothesis.

## V. Implementation Details

A hypothesis is obtained locally by analyzing neighboring surface segments in the image. The way of obtaining hypotheses is efficient and robust to occlusion, especially when several objects are presented in a scene, and each object is partially visible. However, local analysis does not guarantee a valid matching.

Given a hypothesis of $s_i \in TRANSFORM(m)$, where $s_i \in s' \subseteq s = \{s_1, s_2, \ldots, s_t\}$, which are the surface segments in the image, and $m \in$ model set $M$, we build an image $P$ from $TRANSFORM(m)$ and then compare $P$ with the input image $O$ to make decisions based on the correlation between $P$ and $O$.

Suppose that each object in the scene has at least $\gamma\%$ area visible, for instance 50%. We compare the image points on $P$ with values in corresponding positions on $O$. If more than $\gamma\%$ of the image points on $P$ can match with input image $O$ within a noise estimate $\delta$, for instance 5%, we can believe that the hypothesis is true and then erase the matching region of $P$ from $O$ such that $O$ does not contain the recognized and localized object for the next iteration. To obtain hypothesis image $P$, we simply use the $z$-buffer technique of computer graphics [5]. This technique is also called the depth-buffer image space algorithm. The algorithm is simple to implement. The performance of the algorithm tends to be constant since, on the average, the number of pixels covered by each surface decreases as the number of surfaces in the view volume increases [5]. The algorithm works in the following way. The hypothesis image $P$ is initialized to the largest representable $z$ value. For each point $(x, y)$ inside $TRANSFORM(m)$, do the following:

1) Compute the depth $z(x, y)$ at $(x, y)$.
2) If $z(x, y)$ is less than the image $P$ value at $(x, y)$, then place $z(x, y)$ into the image $P$ at $(x, y)$.

The algorithm records the smallest $z$ encountered for each $(x, y)$. The order of surfaces of the model has no effect on the resulting image. The hidden surfaces have larger $z$ values, which are replaced by visible surfaces' $z$ values.

We have implemented this approach as a test vision system *free-form object recognition and localization* (FORL) and tested this on synthesized range images. FORL was implemented in C on a Sun-3/160C Workstation.

The vision process uses information in the knowledge base about surface shapes and the CAD database to recognize and localize 3-D free-form objects. Range images serve as inputs to the vision process. The vision system identifies objects and their transformations with respect to the ideal positions of the CAD models. Range segmentation, which was discussed in Section IV-C is carried out next. This is followed by the surface matching evaluation procedure described in Section IV-D. After this, a hypothesis is provided for verification using the technique introduced in Section IV-E. Finally, a synthetic image is produced from the hypothesis and verified as

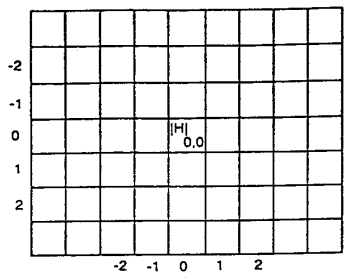


Fig. 9. Magnitude of mean curvature layer.

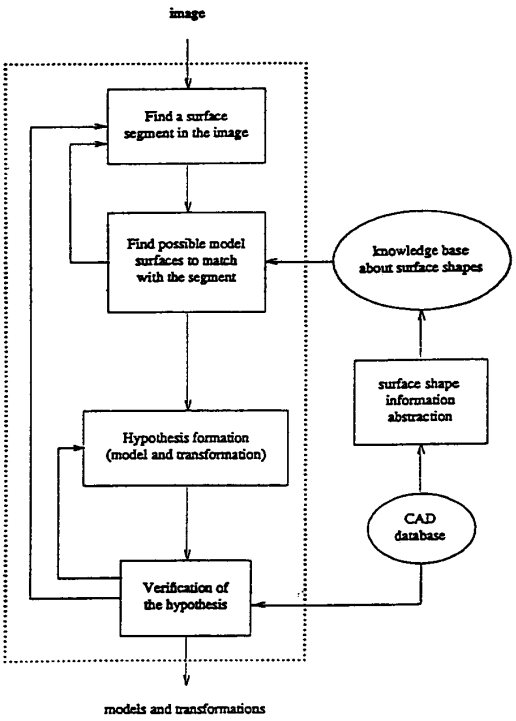


Fig. 10. Flow diagram of FORL.

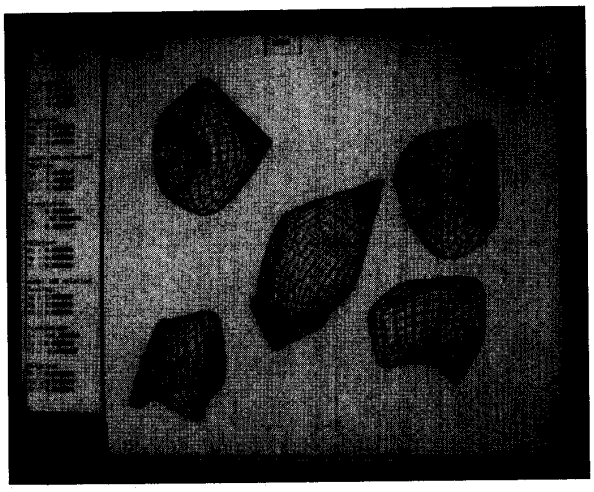

Fig. 11. Objects in the CAD database.

Fig. 12. More objects in the CAD database.

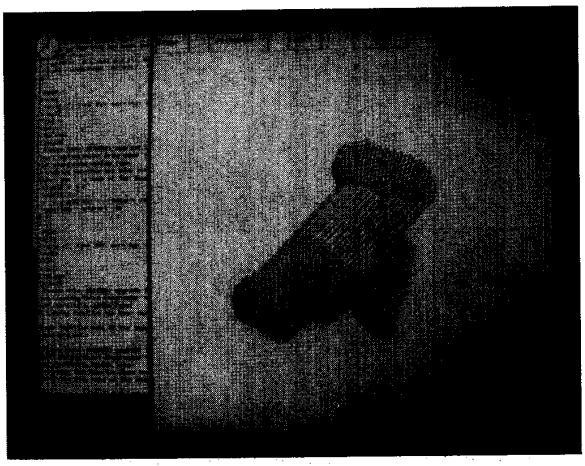

Fig. 13. Contour of the range image.

shown in Section IV-F. The CAD database is the source of the object models. Surface shape information is abstracted from CAD models and then stored in the knowledge base. The following is one of the examples of recognizing and localizing multiple 3-D free-form objects in FORL. Figs. 9–11 show the objects in the CAD database. We arbitrarily selected three objects and arbitrarily arranged them in 3-D space. Fig. 12 shows the contour of the range image.

Fig. 13 shows that one object is recognized and localized. When the first recognized object is erased, the contour of the image is shown in Fig. 14. Fig. 15 shows that the next object is recognized and localized. Again, the contour of the image is shown in Figs. 16 and 17 when the second object is erased. Fig. 18 shows the recognition and localization of the last object.

## VI. Conclusions

We have designed an effective method of surface characterization of 3-D objects using surface curvature properties and developed an efficient approach to recognizing and localizing multiple 3-D free-form objects (*free-form object recognition and localization* (FORL)). Our approach is surface based, which is not sensitive to noise and occlusion, forms hypotheses by local analysis of surface shapes, does not depend on the visibility of complete objects, and uses

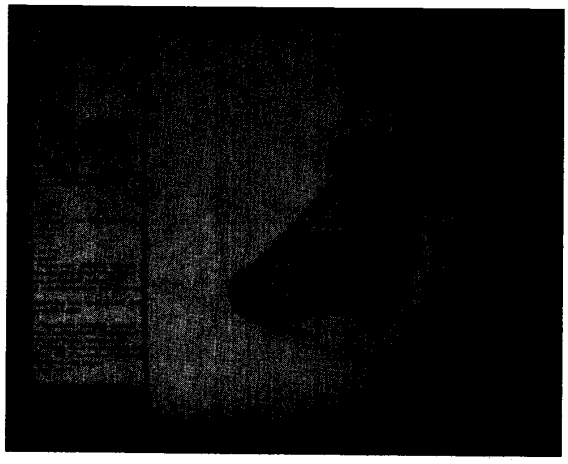

Fig. 14. One object is recognized and localized.

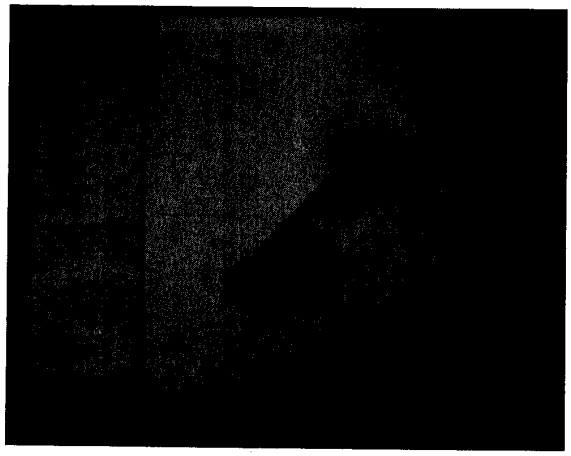

Fig. 15. Recognized and localized object is erased.

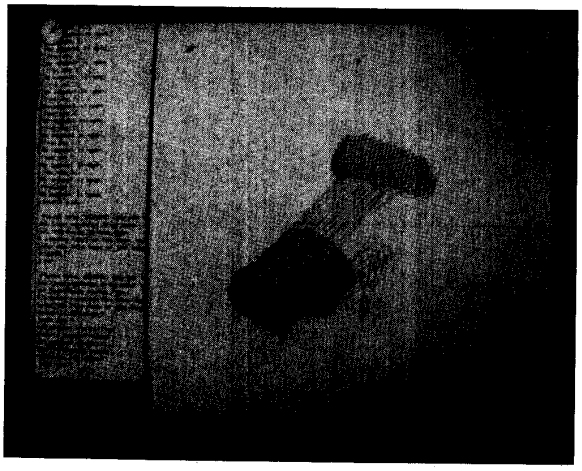

Fig. 16. Second object is recognized and localized.

information from CAD database in recognizing and localization. We have developed a knowledge representation scheme for describing free-form surface shapes. The data structure and procedures are well designed so that the knowledge leads the system to intelligent behavior. Knowledge about surface shapes is abstracted from CAD models to direct the search in verification of vision hypotheses. The knowledge representation used eases processes of knowledge acquisition, information retrieval, modification of knowledge base, and reasoning for solution. Future research should be on combining surface-based recognition and edge-based systems. Our approach is very suitable for parallel processing to increase efficiency [11]. Developing parallel algorithms for our approach to 3-D vision is another future research direction.

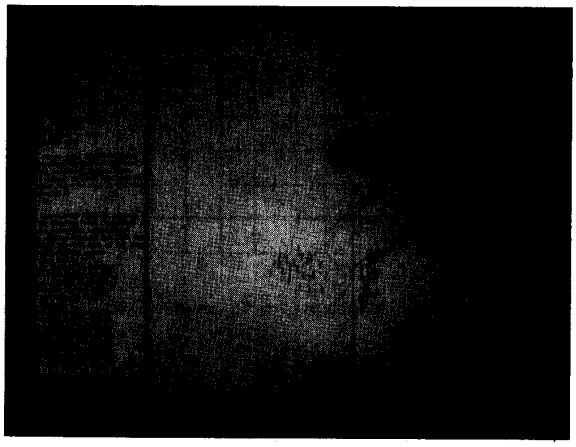

Fig. 17. Second recognized and localized object is erased.

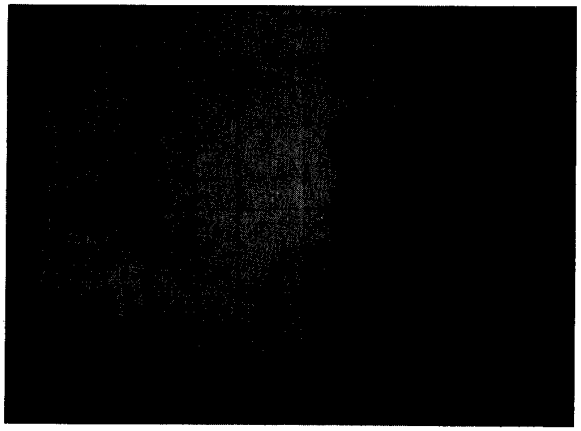

Fig. 18. Last object is recognized and localized.

## ACKNOWLEDGMENT

The authors are grateful to all the reviewers for their critical comments, which improved the presentation of this paper. The authors are also thankful to J. Mundy and A. Jain for their comments on the paper.

## REFERENCES

[1] P. J. Besl, and R. C. Jain, "Three dimensional object recognition," *ACM Comput. Surveys,* vol. 17, no. 1, pp. 75–145, 1985.

[2] W. Wang, "Model-based three dimensional object recognition and localization using properties of surface curvatures," Ph.D. dissertation, Dept. of Comput. Sci., Louisiana State Univ., Dec. 1989.

[3] R. C. Bolles and P. Horaud, "3DPO: A three-dimensional part orientation system," *Three-Dimensional Machine Vision*, 1987.
[4] B. R. Dewey, *Computer Graphics for Engineers*. new York: Harper and Row, 1988.
[5] J. D. Foley and A. Van Dam, *Fundamentals of Interactive Computer Graphics*. Reading, MA: Addison-Wesley, 1984.
[6] P. J. Flynn and A. K. Jain, "CAD-based computer vision: From CAD models to relational graphs," *IEEE Trans. Patt. Anal. Machine Intell.*, vol. 13, no. 2, pp. 114–132, Feb. 1991.
[7] K. T. Gunnarsson and F. B. Prinz, "CAD model-based localization of parts in manufacturing," *Comput.*, pp. 66–74, Aug. 1987.
[8] C. -C. Hsiung, *A First Course in Differential Geometry*. New York: Wiley, 1981.
[9] P. J. Besl and R. C. Jain, "Segmentation through variable-order surface fitting," *IEEE Trans. Patt. Anal. Machine Intell.*, vol. 10, no. 2, pp. 167–192, 1988.
[10] M. M. Lipschutz, *Differential Geometry*. New York: McGraw-Hill, 1969.
[11] R. A. Brooks, *Model-Based Computer Vision*. UMI Research, 1984.
[12] L. G. Roberts, "Machine perception of three-dimensional solids," in *Opt. Electro-Opt. Inform. Processing* (J. P. Tippet *et al.* Eds.). Cambridge, MA: MIT Press, 1965.
[13] I. Vaisman, *A First Course in Differential Geometry*. New York: Marcel Dekker, 1984.
[14] B. C. Vemuri and J. K. Aggarwal, "Localization of objects from range data," *IEEE Conf. Comput. Vision Patt. Recogn.*, 1988, pp. 893–898.
[15] A. K. C. Wong, S. W. Lu, and M. Rioux, "Recognition and shape synthesis of 3-D objects based on attributed hypergraphs," *IEEE Trans. Patt. Anal. Machine Intell.*, vol. 11, no. 6, pp. 643–649, 1989.
[16] W. Wang, S. S. Iyengar, and L. M. Patnaik, "Memory-based reasoning approach for pattern recognition of binary images," *Patt. Recogn.*, vol. 22, no. 5, pp. 505–518, 1989.
[17] W. E. L. Grimson and T. Lozano-Perez, "Recognition and localization of overlapping parts from sparse data," in *Three-Dimensional Machine Vision* (T. Kanade, Eds.), 1987.
[18] W. E. R. Grimson and D. P. Hutenlocher, "On the verification of hypothesized matches in model-based recognition," *IEEE Trans. Patt. Anal. Machine Intell.*, vol. 13, no. 12, pp. 1201–1213, Dec. 1991.
[19] B. Bhanu and L. A. Nuttall, "Recognition of 3-D objects in range images using a butterfly multiprocessors," *Patt. Recogn.*, vol. 22, no. 1, pp. 49–64, 1989.

# Surface Shape Reconstruction of a Nonrigid Transparent Object Using Refraction and Motion

Hiroshi Murase

***Abstract*—The appearance of a pattern behind a transparent, moving object is distorted by refraction at the moving object's surface. This paper describes an algorithm for reconstructing the surface shape of a nonrigid transparent object, such as water, from the apparent motion of the observed pattern. This algorithm is based on the optical and statistical analysis of the distortions. It consists of the following parts: 1) extraction of optical flow, 2) averaging of each point trajectory obtained from the optical flow sequence, 3) calculation of the surface normal using optical characteristics, and 4) reconstruction of the surface. The algorithm is applied to both synthetic and real images to demonstrate its performance.**

***Index Terms*—Computer vision, image recovery, motion analysis, optical flow, surface reconstruction.**

Manuscript received January 3, 1990; revised February 3, 1992. Recommended for acceptance for by Associate Editor N. Ahuja.

The author is with NTT Basic Research Labs, Tokyo, Japan.

IEEE Log Number 9200155.

## I. Introduction

One of the primary tasks of a computer vision system [1] is to capture 3-D information, such as surface orientation, from 2-D images. This task is usually difficult. However, if some cues are known about the scene, such as stereopsis (e.g., [2], [3]), shading (e.g., [4], [5]), contour (e.g., [6], [7]), texture (e.g., [8], [9]), and motion (e.g., [10]–[18]), 2-D images may provide information about the surface. This information is first converted into local surface orientation [19] and then into the shape of the surface. This correspondence proposes a method for reconstructing the surface shape of an undulating transparent object, such as a water surface, using the cues of refraction and motion. Surface orientation is made from the moving (apparent distortion) of patterns viewed through the object. This approach is similar to the method called shape from motion. The previous work in shape from motion involves reconstructing a 3-D structure from movement of several points on the object based on an assumption of rigidity (e.g., Ullman *et. al.* [10]–[17]). The method described in this paper does not use the rigidity assumption but uses physical characteristics such as optical laws and statistical motion features of the object. In addition, the method uses points on refracted images rather than points on the object.

In this method, the objects should have the following two characteristics: 1) They should be transparent with a refraction index not equal to unity, and 2) their surface shape should be deformed around an average surface whose shape is known. To clarify the essence of the idea, an example of a water surface with waves [20], [21], like the surface of a pool or river, is used. Because of refraction at the water surface, the observed pattern of objects under water with waves appears to be moving. Note that human beings can perceive the surface shape from the observed moving pattern. In this case, the above two characteristics correspond to the following: 1) Water is transparent and has a refraction index of 1.33; 2) the average surface is usually a plane whose surface normal is vertical, and a wave can be regarded as a deformation from the average shape. The goal here is to reconstruct the shape of the water surface from deformed images observed through the waving water and, in so doing, recover the original pattern under the water. The original pattern under the water is assumed to be unknown.

This method has two main original ideas. First, it can be considered to be the inverse operation of the ray-tracing approach. The ray-tracing technique is a common method in the field of computer graphics. It is based on an optical law such as a refraction law (Snell's law) or a reflection law and is used to synthesize the images from the given model (shapes of the objects). For example, Ts'o [22] synthesized ocean wave images using physical characteristics and ray-tracing techniques.

Second, this method uses the idea that the pattern observed through the undulating surface is deformed around the pattern observed through the average surface. In the simplest case of the top view and orthographic projection, the average coordinate of the trajectory of a certain point on the distorted pattern corresponds to the the point observed through the static flat water surface. This means that the average position of the point becomes the position observed when there is no water.

The algorithm consists of the following four parts:

1) Optical flow is calculated from an image sequence observed through the moving water surface. Here, "optical flow" refers to point-to-point correspondence between two succeeding image frames. A trace of the optical flow becomes the trajectory of