Computer Vision and Image Understanding **81,** 166–210 (2001)
doi:10.1006/cviu.2000.0889, available online at http://www.idealibrary.com on IDEAL®

A Survey Of Free-Form Object Representation and Recognition Techniques

Richard J. Campbell and Patrick J. Flynn

Signal Analysis and Machine Perception Laboratory, Department of Electrical Engineering, Ohio State University, 205 Dreese Laboratory 2015 Neil Avenue, Columbus, Ohio 43210-1272
E-mail: campbelr@ee.eng.ohio-state.edu, flynn@ee.eng.ohio-state.edu

Received March 6, 2000; accepted November 9, 2000

Advances in computer speed, memory capacity, and hardware graphics acceleration have made the interactive manipulation and visualization of complex, detailed (and therefore large) three-dimensional models feasible. These models are either painstakingly designed through an elaborate CAD process or reverse engineered from sculpted prototypes using modern scanning technologies and integration methods. The availability of detailed data describing the shape of an object offers the computer vision practitioner new ways to recognize and localize free-form objects. This survey reviews recent literature on both the 3D model building process and techniques used to match and identify free-form objects from imagery.

1. INTRODUCTION

Computer models of *free-form* objects are a key element of many interesting applications involving graphics and visualization (e.g., virtual world design and environment modeling for semi-autonomous navigation [54]). In particular, techniques for the automatic recognition of 3D objects have become increasingly sophisticated and the use of free-form object models in such systems is now seeing focused attention from researchers in the computer vision community. Moreover, recent years have seen work in the graphics and vision communities intersect in the area of free-form object modeling. Graphics practitioners, faced with the labor-intensive process of redesigning complex objects on CAD systems, would prefer to use vision techniques to assemble multiple views of a complex object together into a model that can be fine-tuned and completed interactively. Vision practitioners, faced with the limited scope of simple geometric object representations, see free-form representations as a *lingua franca*, a universal language, for future object recognition systems.

In this paper we will survey the recent work done to represent and recognize free-form objects. Included in the discussion about representation is a definition of free-form

1077-3142/01 $35.00

objects (Section 2), coverage of methods used to represent object geometry (Section 3), and a description of some current techniques proposed in the computer vision and graphics literature to develop a model of the object from 3D (range) image data (Section 4). Specific techniques covered include image data registration (Section 4.1), integration (Section 4.2), and model optimization (Section 4.3). The second major portion of the survey addresses techniques used to identify free-form objects present in data gathered from intensity or range image scanners (Section 5). This includes appearance-based (Section 5.1), intensity contour-based (Section 5.2), and surface feature-based techniques (Section 5.3). The survey concludes with some speculation on areas ripe and deserving further research (Section 6).

2. FREE-FORM OBJECTS AND SURFACES: DEFINITIONS, PROPERTIES, AND ISSUES

Definitions of free-form surfaces and objects are often intuitive rather than formal. Synonymous adjectives include "sculpted," "free-flowing," "piecewise-smooth," or "piecewise-C^n" for some desired degree of continuity n. Often, "free-form" is a general characterization of an object whose surfaces are not of a more easily recognized class such as planar and/or natural quadric surfaces. Hence, a *free-form object* is often assumed to be composed of one or more nonplanar, nonquadric *surfaces* ("free-form *surface*"). A roughly equivalent characterization was provided by Besl [11]: "a free-form surface has a well defined surface normal that is continuous almost everywhere except at vertices, edges and cusps." Dorai and Jain [31], Besl [11], and Stein and Medioni [105] cite sculptures, car bodies, ship hulls, airplanes, human faces, organs, and terrain maps as being typical examples of free-form objects. Specifically excluded from this class of object by these authors are statistically defined shapes like textures and foams, infinitely detailed objects possessing self-similarity that are best described using fractal models, and nonorientable surfaces such as Moebius strips and Klein bottles.

Despite the different application contexts of free-form object models in vision and graphics, some criteria apply (or should apply) to representations regardless of domain. In an early survey on object representation, Brown [16] lists ambiguity, conciseness, and uniqueness as some mathematical properties of object representation. *Ambiguity* measures the representation's ability to completely define the object in the model space; this is sometimes referred to as *completeness* of the model description in the computer vision literature. Conciseness represents how efficiently (compactly) the description defines the object. Finally, *uniqueness* is used to measure if there is more than one way to represent the same object given the construction methods of the representation. If the representation is unambiguous and unique then there is a one-to-one mapping from the object to the representation.

The importance of these mathematical properties to the object representation strategy depends on the application context. In the case of object recognition applications, completeness and compactness are often sacrificed in favor of efficiency [36]. The pragmatic issue of performance often makes such compromises appropriate. This highlights the application dependence on the use of *complete* vs *discriminatory models*. Discriminatory models are most often used in object recognition because they can be designed to capture the details that differentiate objects from one another efficiently. These representations are designed to be efficient for the task of matching, and not as a complete description of an object's geometry.

By contrast, some computer graphics applications require complete models to accurately render a realistic synthetic image of the objects described. When rendering, the key attributes relate to realism (visual fidelity). One common method to improve realism of 3D rendering is to increase polygon density and the quality of the accompany texture maps for the models used in rendering.

In a computer vision context, the object model is designed for use in the specific vision application (such as navigation, recognition, or tracking), and visual fidelity may not be a criterion of interest. Efficient execution of the application is almost always of interest, and this may favor multiple redundant representations. In object recognition, *saliency* is an important feature of object representations; the qualities (geometric or otherwise) that allow objects to be discriminated from one another must be easily obtained. It is tempting to assume that salient features are generic, i.e., that saliency is captured by the cardinality or parameters of a particular predefined geometric or photometric feature. As the mix of models changes in a dynamic database, the saliency of some features and models will change, requiring new features to be found or more advanced techniques to be developed that deal with the lack of uniqueness.

The *locality* of an object representation may be of interest in applications of recognition in the presence of occlusion. A representation that explicitly reveals local geometrical structure may be characterized as "occlusion tolerant", hence better suited to such applications. Unfortunately, representations that are chiefly local are generally verbose. This further motivates the use of multiple representations in applications where requirements conflict; this adds a burden, namely the need to maintain consistency between representations.

With the foregoing definitions of free-form surfaces and objects as background, some object recognition systems working with intensity imagery have employed (or assumed) such surfaces and objects for some time. Indeed, some of these systems make no explicit assumption about a class of allowable object shapes. In particular, image-based recognition systems (which employ no specific object model representation; appearance-based recognition is an example) process images of free-form objects just as they would process an image of a geometrically simpler object. Recognition decisions in such systems are based on the distribution of intensities in prototypical views of the object rather than on an object-centered representation *per se*.

The history of model-based 3D object recognition techniques, by contrast, shows a progression in the complexity of object models employed, from simple polyhedra to natural quadrics to various free-form-like object representations such as superquadrics. The choice of a representation must be accompanied by robust techniques for extracting compatible features from both object models and input images, so that recognition can be performed. The potential lack of simple features like crease edges or linear silhouette edges makes this feature identification task difficult. Nonetheless, a variety of creative approaches to this problem have been proposed and implemented, and a goal of this survey is to highlight many of these techniques.

3. GEOMETRIC DESCRIPTIONS OF 3D OBJECTS

This section provides an overview of the popular techniques for representing the geometry (i.e., the 3D shape) of 3D surfaces and objects, with a focus on those representations that are general enough to be labeled "free-form."

3.1. Complete Mathematical Forms

The representations discussed here are *complete* in that the geometric description is explicit, the entirety of a surface or object is described, and hence that synthetic images of the object in an arbitrary pose can be generated when the geometry description is coupled with a view specification.

3.1.1. Parametric forms. Parametric surfaces are widely used in computer aided design and manufacturing (CADM) and other object modeling systems for the following reasons:

1. They are mathematically complete.
2. They are easily sampled.
3. They facilitate design (objects can be designed in terms of patches whose continuity can be controlled at the joins).
4. Their representational power is strong (they can be used to represent complex object geometries).
5. They can be used to generate realistic views.
6. Methods for generating parametric representations for data points taken from existing objects have been developed [30, 35].

A generic parametric form for a 3D surface (a 2D manifold embedded in 3D) is

$$S(u, v) = \begin{bmatrix} x = f(u, v) \\ y = g(u, v) \\ z = h(u, v) \end{bmatrix}.$$

The three functions $f(u, v)$, $g(u, v)$, and $h(u, v)$ have as arguments the two parametric variables (u, v). Without loss of generality the domain of (u, v) can be restricted to be the unit square $[0, 1] \times [0, 1]$.

The most common parametric formulation is the nonuniform rational B-spline (NURBS), Fig. 1, which are defined by the specialized parametric form

$$S(u, v) = \sum_i \sum_j B^{\mathrm{h}}_{i,j} N_{i,k}(u) M_{i,l}(v),$$

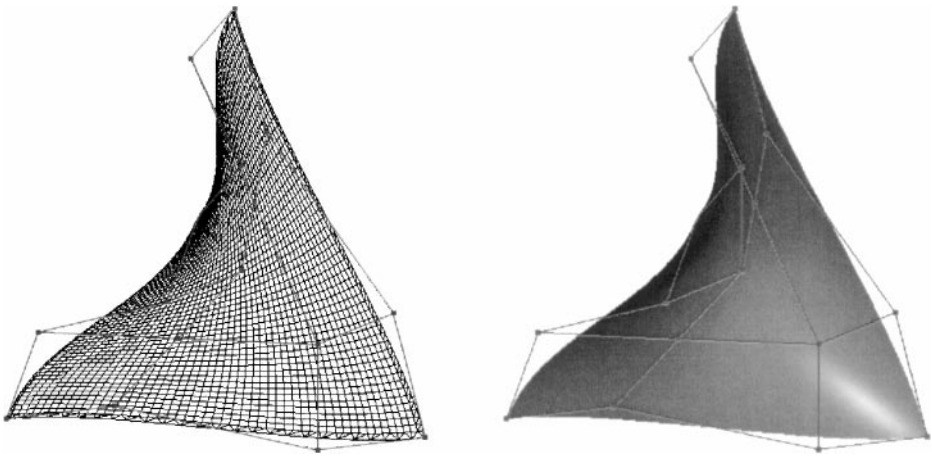

FIG. 1. A Curved NURBS patch depicted as a mesh and as a shaded surface. The control polyhedron is visible in both images.

where $N_{i,k}(u)$ and $M_{i,l}(v)$ are the B-spline basis functions of order k and l, and $B^{\mathrm{h}}_{i,j}$ are the homogeneous coordinates of the control points, respectively [64]. Hence, NURBS are a tensor-product surface form. It has been shown that natural quadrics (such as spheres, cylinders, and cones) admit exact representation as NURBS; the available homogeneous coordinate makes this representation quite flexible [35]. Figure 1 shows a curved NURBS patch as a parametric mesh and a shaded surface, including the positions of vertices in its control point mesh.

This common descriptive form has been considered for use in computer vision systems, but Besl [11] explains why parametric representations are not widely used. In particular, it is difficult to make a surface defined on a parametric rectangle fit an arbitrary region on the surface of an object; this necessitates the use of trimming curves, which are not always unique and not generally detectable in imagery. Moreover, the homogeneous control points are also not easily detectable nor unique. The completeness of parametric forms makes them useful as a source of an initial object specification, from which a polygonal mesh or other representations can be generated and employed in a vision system.

3.1.2. Algebraic implicit surfaces. A surface can be defined implicitly as the zero-set of an arbitrary function f:

$$\mathcal{S} = \{(x, y, z) | f(x, y, z) = 0\}.$$

Figure 2 depicts an implicit quartic surface. If f is polynomial and a set of samples from a single surface of this form is available, well-established fitting procedures can be used to estimate the polynomial coefficients [108]. These coefficients are not generally invariant to pose transformations, although mathematical invariants can be obtained from some forms [106]. Recent work in this area has included.

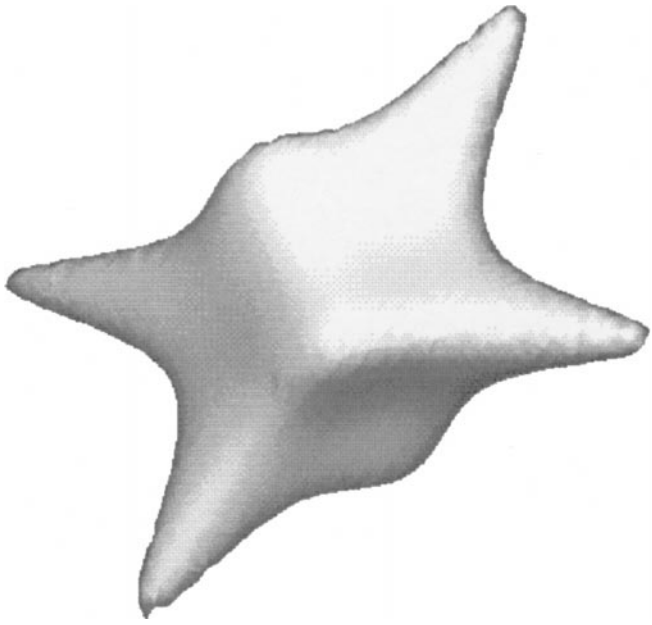

FIG. 2. Implicit model defined by algebraic equation $f(x, y, z) = 2x^4 - 3x^2y^2 + 3y^4 - 3y^2z^2 - 2x^3z + 6z^4 - 1 = 0$.

1. finding the exact distance between an implicit surface and 3D points sampled from it [107],
2. using bounded versions of algebraic curves and surfaces to control the fitting process [109], and
3. preserving a known topology in the fitted surface [107] (i.e., if the data is assumed to be from a surface of genus zero then the fitted surface will be of genus zero).

Surface fitting techniques have been used to segment image scenes [108] and have also been used as a preprocessing step in the recognition of objects [106].

The class of objects that can be described by a single algebraic surface is limited by the stability of the fitting process. In practice, fitting a surface to an order greater than 4 can produce surfaces whose shape matches poorly to the object from which the data were obtained. In order to model more complex objects, patches of implicit surfaces must often be used [4, 107].

3.1.3. Superquadrics. A superquadric $S(\eta, \omega)$ is a volumetric primitive [6, 100] with the simple implicit and parametric forms

$$S(\eta, \omega) = \begin{bmatrix} x(\eta, \omega) \\ y(\eta, \omega) \\ z(\eta, \omega) \end{bmatrix} = \begin{bmatrix} a_1 \cos^{\epsilon_1}(\eta) \cos^{\epsilon_2}(\omega) \\ a_2 \cos^{\epsilon_1}(\eta) \sin^{\epsilon_2}(\omega) \\ a_3 \sin^{\epsilon_1}(\eta) \end{bmatrix}, -\frac{\pi}{2} \le \eta \le \frac{\pi}{2}, -\pi \le \omega \le \pi,$$

$$S(x, y, z) = \left[\left[\left(\frac{x}{a_1} \right)^{\frac{2}{\epsilon_1}} + \left(\frac{y}{a_2} \right)^{\frac{2}{\epsilon_2}} \right]^{\frac{\epsilon_2}{\epsilon_1}} + \left(\frac{z}{a_3} \right)^{\frac{2}{\epsilon_1}} \right]^{\epsilon_1} = 0.$$

Pentland [86] and Solina and Bajcsy [100] showed that with the addition of tapering, twisting, and bending deformations to the superquadric models, a variety of free-form objects can be modeled (see Fig. 3 for an example). Such models capture only with great difficulty the fine detail that differentiates similar objects like human faces from one another. The ability of the superquadric to model the coarse shape of an object has been used to determine geometric classes (or Geons) [91, 34].

Any object recognition system employing superquadric models as a primary representation must solve two key problems in its design. First, a mechanism for segmenting the input image data (typically range data) into patches, each of which lies within a single superquadric primitive, must be developed. Secondly, a robust technique for fitting superquadric models to the sensed data must be available. Gupta and Bajcsy [44], Pentland [86], Solina and Bajcsy [100], Raja and Jain [91], and Dickinson *et al.* [34] have all addressed facets of this problem.

Superquadrics have been used to recognize a large class of objects and have even been used to build composite parts (unions of several superquadrics), but they lack the descriptive ability to effectively capture local surface shape changes on free-form objects. Ayong-Chee *et al.* [3] demonstrated a hybrid approach to recognition using the parameters of a superquadric to define object shape classes. These classes are used to remove possible object matches based on their gross shape. Then more computationally expensive techniques can be used to match the fine details of objects within a class.

3.1.4. Generalized cylinders. Generalized cylinders are defined by a space curve $\mathbf{A}(s)$ (representing the axis of the primitive) and a cross-section contour $\mathbf{C}(s, \theta)$ defined in the

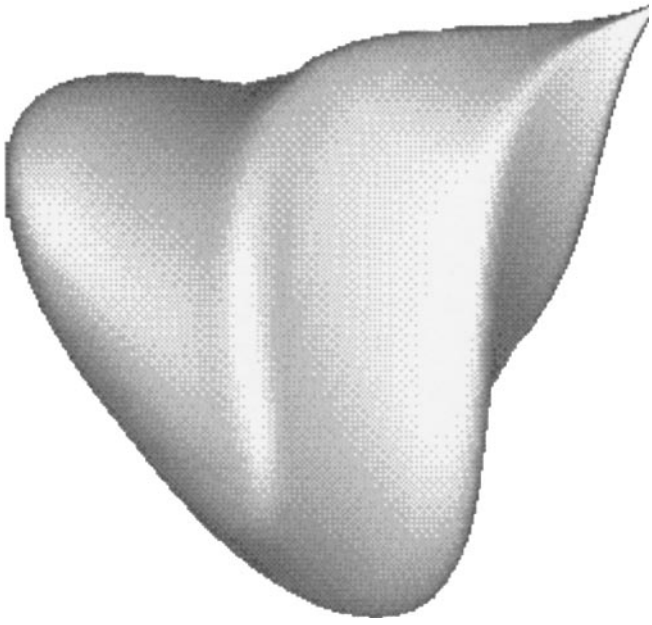

FIG. 3. Deformed superquadric. The superquadric ($a_1 = a_2 = a_3 = 1$ and $\epsilon_1 = 1.0, \epsilon_2 = 0.3$) is tapered along the z axis

$$\left(\begin{bmatrix} x_t(\eta,\omega) \\ y_t(\eta,\omega) \end{bmatrix} = \left(\frac{z(\eta,\omega)+1}{2}\right)\begin{bmatrix} x(\eta,\omega) \\ y(\eta,\omega) \end{bmatrix}\right),$$

then twisted about the z axis using $\theta = \pi(1 - z(\eta,\omega))$,

$$\left(\begin{bmatrix} x_{tt}(\eta,\omega) \\ y_{tt}(\eta,\omega) \end{bmatrix} = \begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix}\begin{bmatrix} x_t(\eta,\omega) \\ y_t(\eta,\omega) \end{bmatrix}\right).$$

plane normal to the axis at s and defining the boundary of the primitive along the axis [1, 81]. Figure 4 shows an example of a free-form object represented by a generalized cylinder. To construct more complex objects (e.g., hierarchies such as animals), multiple generalized cylinders are used to represent their individual parts.

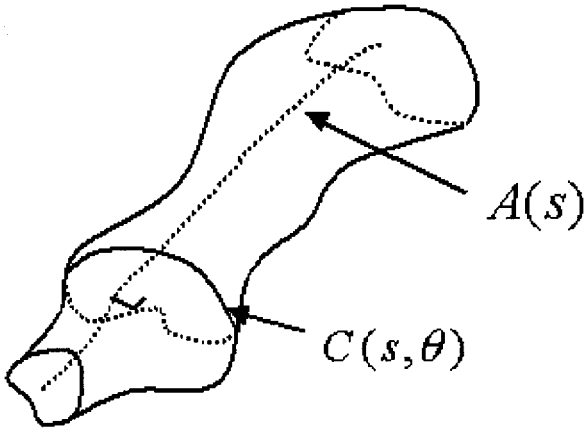

FIG. 4. Generalized cylinder.

Generalized cylinders are particularly attractive for representing elongated shapes where an axis is easy to define. In this case the axis of the primitive often provides an intuitive method to conceptualize the design of a object and a method of reliably recovering useful statistics about the shape of the object.

On the other hand, some shapes are not easily described using generalized cylinders. In such cases, it may be difficult (or impossible) to define an axis whose cross sections contain only one closed contour. It is possible to extend the representation to handle these and other cases that appear, but it may not prove to be the most useful representation of the object with respect to design and discrimination.

3.1.5. Polygonal meshes. A popular representation for 3D objects is the polygonal mesh. A shared vertex-list notation is common for such representations. Accordingly, an object is defined by a pair of ordered lists

$$\mathcal{O} = \langle \mathcal{P}, \mathcal{V} \rangle,$$

where $\mathcal{V} = \{\mathbf{v}_1, \ldots, \mathbf{v}_{N_v}\}$ is a list of N_v three-dimensional vertices $\mathbf{v}_i = (x_i, y_i, z_i)^{\mathrm{T}}$, and $\mathcal{P} = \{\mathbf{p}_1, \ldots \mathbf{p}_{N_p}\}$ is a list of polygons, each specified as a list of vertex indices: $\mathbf{p}_i = \{v_{i,1}, \ldots, v_{i,nv_i}\}$.

If $nv_i = 3$ for all i, the mesh consists strictly of triangles. The guaranteed convexity of triangles allows simpler rendering algorithms to be used for the generation of synthetic images of models [38]. A variety of techniques (commonly called *polygonization* methods) exists for generating polygonal mesh approximations from other geometric primitives (such as implicit surfaces [82], parametric surfaces [65], and isosurfaces in volume data [70]).

Polygonal meshes have a long history in computer graphics, but have also become increasingly popular as an object representation in computer vision. This increase in popularity is due to several factors including advances in computer storage capacity and processing power and a modest increase in the popularity of dense range sensors, which produce rectangular arrays of 3D points that can easily be triangulated into meshes. Meshes can faithfully approximate complex free-form objects to any desired accuracy given sufficient space to store the representation. Figure 5 shows (in wireframe and shaded renderings) a polygonal mesh representing a human foot bone. With the decreasing cost of computer memory, even very dense meshes are becoming practical. The expanding market for virtual environments in entertainment and geometric design has also played a role in promoting the mesh as a universal language for object representation.

Polygonal meshes have limitations. While they are a faithful representation, they are also approximate and scale-dependent. Any higher-level surface characterization must be

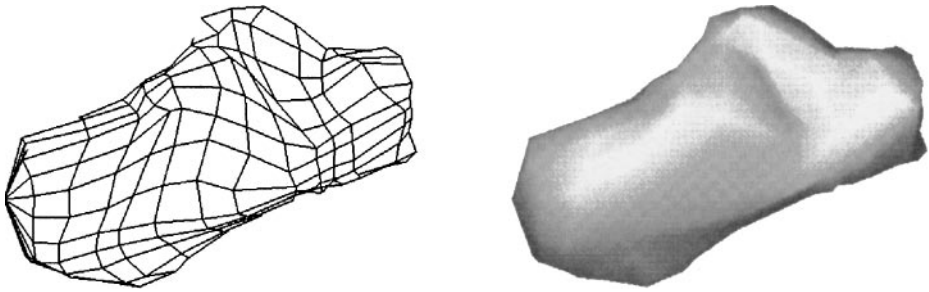

FIG. 5. A coarse polygonal model of the Calcaneus foot bone. The left side is the underlining wire mesh, while the right side shows a Phong-shaded visualization of the model.

explicitly maintained along with the mesh. The required resolution (density) of the mesh may vary between (or within) applications. A strong subarea of research in computer vision and graphics is the study of approaches to coarsen dense meshes, or refine coarse meshes, in response to application requirements. This topic, as well as a variety of approaches to the construction of models from multiple views, is discussed in Section 4.

3.2. Dynamic Objects

Machine vision and/or object modeling systems must sometimes accommodate objects with time-varying shape. In this survey we mention only a few relevant papers on nonrigid shape, limiting the discussion to objects with articulated or deformable shapes. A comprehensive survey of either of these areas is outside the scope of this paper, but would be a valuable service to the community.

3.2.1. Articulated objects. In many cases dynamic objects can be decomposed into a set of rigid parts connected together by joints. This mechanical perspective on both machines and biological forms can closely approximate a large number of dynamic objects. For biological forms whose support structure (bone, cartridge, exoskeleton) is rigid, the primary contributor to change in the shape and appearance of the object is the relative motion at the joints between parts of the form (body, limbs, or appendages).

Articulated objects are typically studied by identifying the rigid parts and parametrizing the relationships between them. As mentioned in the previous sections, superquadric models and generalized cylinders have been used to recover and identify parts of articulated objects. Another reason to study articulated objects is to determine the purpose (function) of the assembly of parts [42]. In the computer vision literature, articulated objects have been studied in a modeling context by Ashbrook *et al*. [2]. Sallam and Bowyer [95] investigated the extension of the aspect graph representation to articulated assemblies.

3.2.2. Deformable objects. Other animals, plants, and machines move in a fluid free-flowing way. Examples of nonrigid motion of object are a swimming jelly fish, wheat stalks blowing in the wind, and a beating heart. These objects surfaces deform as they move and interact with their environment.

To understand and model the changes in a deformable object's surfaces the properties of object materials become more important. One method of doing this is to utilize fienite element models (FEM) to discretize the object into small connected elements [97] whose properties and environmental pressures can be used to predict shape. Another aspect of deformable modeling is to quantify regular motions like those of a beating heart [85]. Analyzing and comparing these regular motions is an important tool for diagnosing and treating defects in heart rhythm and pumping action. Delingette *et al*. [28] studied deformable object shape modeling, and related work is addressed in Section 5.3.

Table 1 summarizes some important dimensions of 3D modeling technique as discussed above. Each strategy is characterized in terms of the accessibility of local shape information, compactness, controllability, completeness, ease of sampling, and ease of construction from sampled data.

4. 3D MODEL CONSTRUCTION AND REFINEMENT

The history of computer-aided 3D design has, until recently, focused on CAD-like approaches to object design. A design engineer would work from a product concept or a

TABLE 1
Properties of Various 3D Object Representations

Representation	Global	Compact	Local control	Complete	Easily sampled	Easily fit
Parametric	yes	yes	yes	yes	yes	no
Implicit	yes	yes	no	yes	no	yes
Superquadric	yes	yes	no	yes	yes	yes
Gen. cylinder	yes	yes	no	yes	yes	no
Mesh	no	no	yes	yes	yes	yes

physical prototype, and encode its geometry using the particular representation tools available in the software. If the geometry of the object did not match the modeling primitives available, cumbersome workarounds and approximations were needed. Recently, a number of experimental and commercial systems composed of versatile range sensors and software have aimed to automatically produce geometric models from physical prototypes of objects. Figure 7 depicts range/texture views and their integration within a particular product of this sort. Depending on the sensor, simple and small mechanical parts, complex assemblies, human bodies, and even environments (e.g., a hallway with doors, obstacles, pipes along the ceiling, etc.) can be scanned and represented. *Reverse engineering* is a popular term for this type of technique.

As noted above, the polyhedral mesh representation has recently become a popular representation for 3D objects. The meshes can be produced through direct polygonization of structured data (e.g., range imagery), polygonization from scattered 3D points, and inflation, deflation, or deformation of an initial spherical mesh to approximate a 3D point set. The polygonization is often simplified to triangles in order to speed up the visualization of reconstructed model and reduce the complexity of the processing algorithms.

The process of reverse engineering the geometry of a 3D model for a part is typically decomposed into the following two or three subtasks, as depicted in Fig. 6:

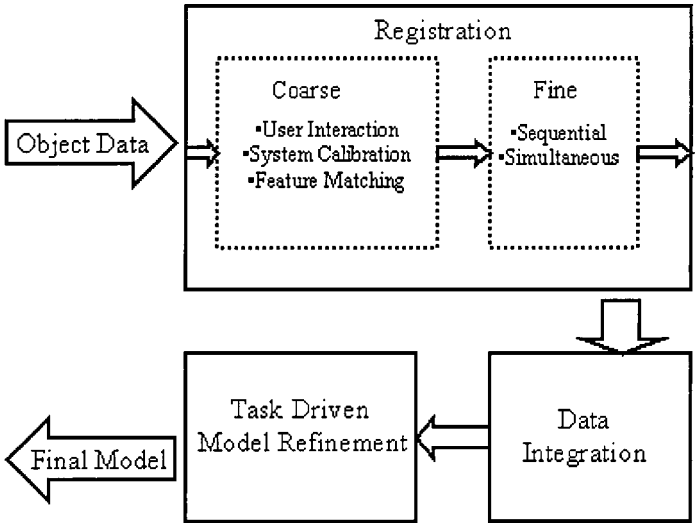

FIG. 6. Flow diagram showing the various steps in the model-building process.

• The first subtask is the registration of all available sets of 3D data to a common (object) coordinate system. This step may be accomplished automatically or semi-automatically.

• The second subtask is the integration of the data to form a single 3D mesh. By "integration," we refer to the merging of independent (but registered) meshes to form a single polygonal mesh without defects such as dangling edges or topology violations.

• The third (optional) task is to optimize the mesh representation for a specific application or use. This often results in a coarsening of the mesh i.e., reduction of the vertex and polygon count) in response to application demands or other constraints. This also could mean changing the object representation to one that is more suited for an application. An example of this is the inclusion of texture map data with the polygonal representation to provide a much more realistic rendering of the object even when the application requires the use of a coarse polygonal mesh to achieve interactivity.

4.1. Registration

Most 3D scanning methods can only capture part of the object at a time. This could be due to limitations in the physical design and/or the technology the scanner is built upon. An example is a structured laser light scanner that works on the principle of triangulation to recover position of surface points on the object. Using this scanner technology parts of an object's surface may not be recoverable because the object itself shadows the structured light from the detector in the sensor. Figure 7 shows an example of data taken from a Minolta Vivid 700 structured light range sensor.[1] In the first four subfigures the result of self-shadowing produces holes in the mesh produced by the scanner.

The construction of a complete object model will require multiple object scans. Since each scan is generally represented in an independent sensor coordinate system, the scan data must be transformed from several different sensor coordinate systems to a single object-centered coordinate system. This step is typically termed *registration* and has received sustained attention in the research community over the past 10 years [17]. During this time a distinction has been drawn between *coarse* registration (wherein the set of interframe registrations are determined to within a few degrees of rotation and a similarly small translational tolerance) and *fine* registration wherein the transformation parameters are fine-tuned. This distinction has been drawn because different techniques are often employed in these two phases.

Most commercial products for model construction employ a rotating turntable upon which the object is placed for sensing. Such systems do not need to solve a coarse registration problem (although pose refinement is still performed to compensate for mechanical variability in the turntable mechanism and calibration errors).

In the model building process a sequence of views $\{\mathbf{V}_1, \ldots, \mathbf{V}_m\}$ is taken of an object O at different viewpoints. The data for each view lie in a local coordinate frame. The goal is a set of transformations $\{T_1^0(\vec{\mathbf{p}}) \ldots T_m^0(\vec{\mathbf{p}})\}$ that will transform the data from the views to a common coordinate frame and minimize an error function.

Let

$$T_i^j(\vec{\mathbf{p}}) = \mathbf{R}\vec{\mathbf{p}} + \vec{\mathbf{d}}$$

[1] Available at http://www.minolta3D.com/.

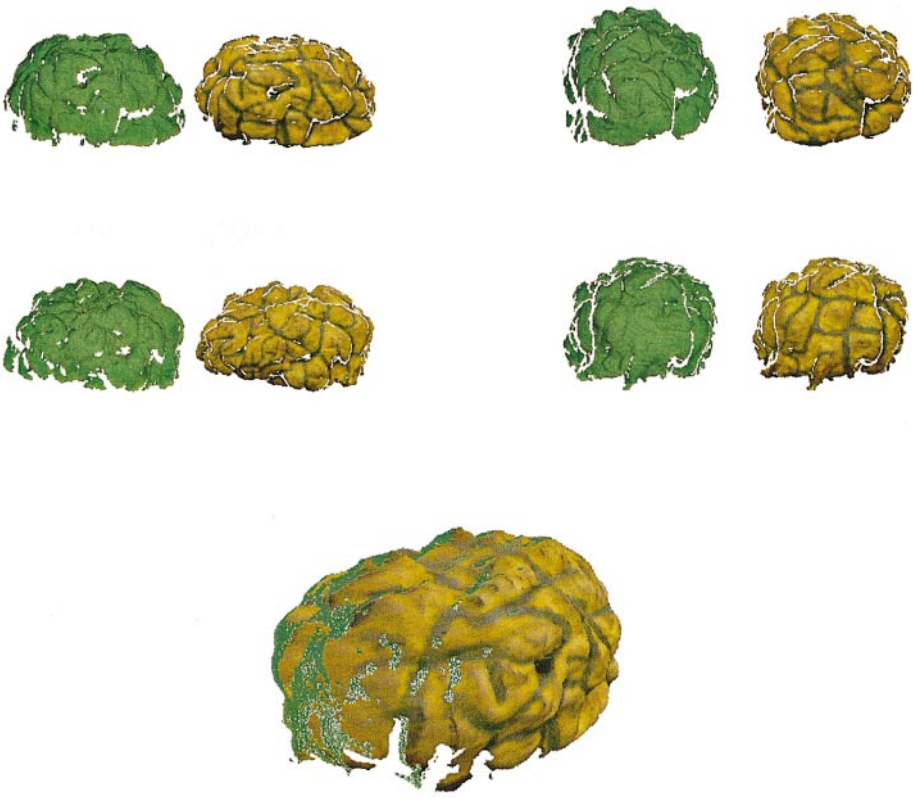

FIG. 7. (top) Three-dimensional scans and registered color images of a *papier mâché* brain, taken 90° apart using a Minolta Vivid 700 noncontact 3D digitizer. (bottom) The registered and merged 3D model with one view's texture mapped on to the merged mesh. The images were generated using Minolta's Vivid software package.

denote a linear transformation of a point $\vec{\mathbf{p}}$ in coordinate frame i to coordinate frame j, where

$$\mathbf{R} = \begin{pmatrix} a_{1,1} & a_{1,2} & a_{1,3} \\ a_{2,1} & a_{2,2} & a_{2,3} \\ a_{3,1} & a_{3,2} & a_{3,3} \end{pmatrix}$$

$$\vec{\mathbf{d}} = \begin{pmatrix} d_1 \\ d_2 \\ d_3 \end{pmatrix}.$$

If the rotation matrix $\mathbf{R}$ is orthonormal then $T_i^j(\vec{\mathbf{p}})$ is a rigid transformation.

4.1.1. Coarse registration. The goal of a coarse registration process is to compute efficiently a set of approximate registration transformations $\{T_1^0(\vec{\mathbf{p}}) \ldots T_m^0(\vec{\mathbf{p}})\}$ to be applied to the data obtained from different views. These transformations are intended to align the data closely enough for a subsequently applied "fine" registration procedure. The coarse registration procedures are designed to improve the chances of the fine registration procedure's convergence to the global optimal solution.

Often the coarse registration is obtained by using a controlled scanning environment. In this case the relationships between the views' coordinate frames $\{T_1^0(\vec{\mathbf{p}}) \dots T_m^0(\vec{\mathbf{p}})\}$ are known a priori from the calibration of the scanner with the mechanics of the sensing environment. The problems with using such well-controlled and accurate scanning systems are objects need to be placed in precise locations in the scene and the automation of the scanner or multiple scanners must be precisely calibrated. These constraints result in more expensive equipment and limitations on the types of objects that can be scanned.

Human interaction can also be used to obtained coarse registration between object views. This a popular method used in some commercial systems. Here, the users are asked to select corresponding points in each of the views $\{\mathbf{V}_1, \dots, \mathbf{V}_m\}$. This manual matching of point correspondence gives the system an initial set of transformations $\{T_1^0(\vec{\mathbf{p}}) \dots T_m^0(\vec{\mathbf{p}})\}$ that can be refined further using optimization techniques.

A more automated approach uses matching techniques (on local surface or albedo data) to recover sets of point correspondences between the views without user interaction. The views $\{\mathbf{V}_1, \dots, \mathbf{V}_m\}$ contain regions of overlap where for any pair of neighboring views $\mathbf{V}_i$ and $\mathbf{V}_j$ there are surfaces that are visible in both views. In these regions of overlap point correspondences can be found by matching similar points in the adjacent views. Let $C_k = (\vec{\mathbf{p}}_{\mathbf{i},\mathbf{k}}, \vec{\mathbf{p}}_{\mathbf{j},\mathbf{k}})$ be a pair of such corresponding points where $\vec{\mathbf{p}}_{\mathbf{i},\mathbf{k}}$ lies on view $\mathbf{V}_i$ and $\vec{\mathbf{p}}_{\mathbf{j},\mathbf{k}}$ lies on view $\mathbf{V}_j$. Many techniques have been developed to match points on arbitrary surface shapes [105, 24, 25, 57, 54, 53]. These methods encode surface geometry around points of interest that are used to identify and match "similar" points. The specifics of the features used and the matching techniques are found in Section 5.3.

For automated registration of multiple views the sets of corresponding points between views are often not sufficient. Since the characteristics that the matching technique uses are not unique, there exists a real possibility that the sets of corresponding points contain mistaken matches [25, 53]. To counteract this, each of the corresponding points between the two views are grouped based on their ability to help form a consistent hypothesis for the registration of the views. Some common consistency tests between pairs of corresponding points C_k and C_l are:

- the interpoint distance $\|\vec{p}_{i,k} - \vec{p}_{i,l}\| - \|\vec{p}_{j,k} - \vec{p}_{j,l}\| < \epsilon_1$,
- angle difference $\|\vec{p}_{i,k} \cdot \vec{p}_{i,l} - \vec{p}_{j,k} \cdot \vec{p}_{j,l}\| < \epsilon_2$, and
- orientation change $\|\vec{n}_{i,k} \times \vec{n}_{i,l} - \vec{n}_{j,k} \times \vec{n}_{j,l}\| < \epsilon_3$ (where $\vec{n}_{i,k}$ is the estimated surface normal at point $\vec{p}_{i,k}$).

The geometrically consistent corresponding points in two viewpoints are then used to produce the transformation $T_i^j(\vec{\mathbf{p}})$ relating views $\mathbf{V}_j$ and $\mathbf{V}_i$. If more than one consistent hypothesis results from the grouping, then each group is used to produce a transformation $T_i^j(\vec{\mathbf{p}})$ and these are compared based on their ability to register the two views $\mathbf{V}_j$ and $\mathbf{V}_i$ with the least error.

4.1.2. Fine registration. Given a set of coarse registrations $\{T_1^0(\vec{\mathbf{p}}) \dots T_m^0(\vec{\mathbf{p}})\}$ relating all the views $\{\mathbf{V}_1, \dots, \mathbf{V}_m\}$, the fine registration task aims to make small changes to the individual transformations $\{T_1^0(\vec{\mathbf{p}}) \dots T_m^0(\vec{\mathbf{p}})\}$ to improve the overall "quality" of the registration. The quality criterion typically rewards a small distance between points in the corresponding surfaces of two "overlapping" views.

4.1.3. Two view optimization. The iterative closest point (ICP) algorithm (Fig. 8) developed by Besl and McKay [10] is widely used for the fine registration task, and uses a

$\mathcal{P}_2(0) = \mathcal{P}_2$
DO
  FOR every point in $\mathcal{P}_2(l)$
    Find the closest point in $\mathcal{P}_1$.
  END FOR
  The closest points form a new point set $\mathcal{Y}(l)$ where the pairs of points
  $\{(\mathbf{p}_1^2, \mathbf{y}_1), \ldots, (\mathbf{p}_{Np_2}^2, \mathbf{y}_{Np_2})\}$ form the correspondences between $\mathcal{P}_1$ and $\mathcal{P}_2(l)$.
  IF registration error between $\mathcal{P}_1$ and $\mathcal{P}_2(l)$ is too large
    Compute registration $T(l)$ between $(\mathcal{P}_2(l), \mathcal{Y}(l))$.
    Apply registration $\mathcal{P}_2(l+1) = T(l) \bullet \mathcal{P}_2(l)$.
  ELSE
    STOP
  END IF
WHILE $\|\mathcal{P}_2(l+1) - \mathcal{P}_2(l)\| >$ threshold

FIG. 8. Iterative closest point(ICP) algorithm.

nonlinear optimization procedure to further align the data sets. Let $\mathcal{P}_1$ be the set of points from the first data set $\{\mathbf{p}_1^1 \ldots \mathbf{p}_{Np_1}^1\}$, where Np_1 is the number of points in the set. Let $\mathcal{P}_2$ be the set of points from the second data set $\{\mathbf{p}_1^2 \ldots \mathbf{p}_{Np_2}^2\}$, where Np_2 is the number of points in the second data set. Since ICP is an iterative algorithm, define an incremental transformation $T(l)$ such that $\mathcal{P}_2(l+1) = T(l) \bullet \mathcal{P}_2(l)$; $T(l)$ reduces the registration error between $\mathcal{P}_1$ and the new point set $\mathcal{P}_2(l+1)$ by moving the points $\mathcal{P}_2(l)$. $\mathcal{P}_2(l) = \mathcal{P}_2$ initializes the system.

The algorithm summarized in Fig. 8 starts by establishing correspondences between the two data set $\mathcal{P}_1$ and $\mathcal{P}_2(0)$. Then the registration error between the two data sets is computed and tested to see if it is within a specified tolerance. In Besl and McKay [10] the registration error was defined with respect to the correspondences between points in the data sets. For each point in $\mathcal{P}_2(l)$ the closest point in $\mathcal{P}_1$ is found. This forms a new point set $\mathcal{Y}(l)$ where point $\mathbf{y}_k$ denotes the closest point in $\mathcal{P}_1$ to the point $\mathbf{p}_k^2(l)$ in $\mathcal{P}_2(l)$. Then the registration error between $\mathcal{P}_1$ and $\mathcal{P}_2(l)$ is

$$E = \frac{1}{Np_2} \sum_{k}^{Np_2} \|y_k - \mathbf{p}_k^2(l)\|^2.$$

If the registration error is too large then the transformation between the views is updated based on a traditional nonlinear optimization technique like Newton's method. The new registration is applied to $\mathcal{P}_2(l)$'s points and tested to see if the optimization has reached a local minimum. This process continues until the registration error falls below a desired quality threshold or sufficiently small changes in the motion between $\mathcal{P}_2(l)$ and $\mathcal{P}_2(l+1)$ are detected, indicating convergence of the solution to a local minima.

The parameter space explored by ICP can contain many local minima. In order to converge to the global minima the initial parameter estimate must be reasonably close to the true value to avoid converging to a nonoptimal solution. This issue motivated the development of the coarse registration procedures discussed above. An alternative to the ICP approach would be to use a different nonlinear optimization algorithm (such as simulated annealing or evolutionary programming) capable of climbing out of local minima.

Zhang [117] has added elements to the general ICP algorithm to handle outliers and occlusion, and to perform general subset–subset matching. The modifications changed the

ICP algorithm's (Fig. 8) routines to select closest points between two point sets $\mathcal{P}_1$ and $\mathcal{P}_2(l)$. Zhang improved the original ICP algorithm by adding some heuristics for whether a particular pair of closest points should be included in the estimation of the new transformation. The new system showed better tolerance for larger uncertainties in the initial guess of pose, erroneous data, and missing data while still managing to converge to the optimal solution.

Chen and Medioni [23] also use an iterative refinement of initial coarse registrations between views to perform fine registration. Instead of minimizing the distance between the closest points in the two point sets, Chen and Medioni employ orientation information. They devised a new least-squares problem where the energy function being minimized is the sum of the distances from points on one view's surface to the tangent planes of another view's surface. This tends to not only minimize the distance between the two surfaces but also match their local shape characteristics [7]. Because of the expense of computing intersections between lines and the discrete surface (generated from the tangent planes) they subsample the original data to obtain a set of "control" points. The set of control points are points located on the surface conforming to a regular grid; furthermore the points are then selected from this subsampling of the surface (regularized grid) that lie in smooth regions on the surface. In these areas of smoothness the surface normal can be calculated more reliably.

Weng *et al*. [18] proposed the use of an "optimal" minimum variance estimate of view registration between two views, assuming uniform i.i.d. Gaussian noise in the sensor's measurement of the depth coordinate; the (x, y) measurement in each imput image is assumed to be noise free. In their discussion of the results they show that for both synthetic and real data the MVE estimate produces lower registration error than the least-squares estimation.

4.1.4. Multiple-view optimization. In the process of forming complete 3D models from multiple images, often more than two viewpoints of data must be combined to form a complete object model. Shum *et al*. [99], Gagnon *et al*. [40], Neugebauer [80], Blais and Levine [14], and Chen and Medioni [23] have examined the propagation of registration error when multiple pairwise estimates of registration transformation are concatenated (e.g., $T_0^4 = T_3^4 \bullet T_2^3 \bullet T_1^2 \bullet T_0^1$). Figure 9a shows graphically, for a series of views $\{\mathbf{V}_0, \ldots, \mathbf{V}_4\}$, one possible way of calculating the pairwise estimates in the registration process. This sequential estimation of the multiple-view registrations can lead to large registration errors between views $\mathbf{V}_0$ and $\mathbf{V}_4$. To minimize this error the registration process can be performed simultaneously for multiple viewpoints, thus minimizing the registration error for the entire object.

The star network topology in Fig. 9b shows the most commonly used relationships between view coordinate frames. In the network the nodes represent each view, while the

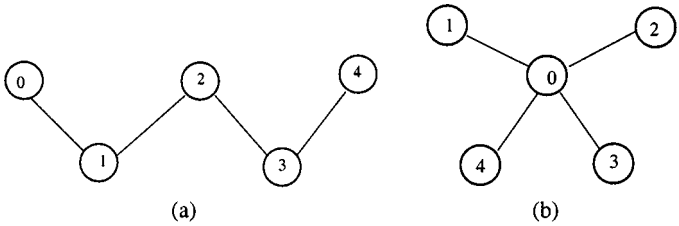

FIG. 9. Two common network topologies used in multiple-view registration.

links represent transformations from one view's coordinate frame to another. The central node (node 0 in the figure) represents the object's coordinate frame. In this topology every view's points $\mathbf{V}_i$ can be transformed into any other view's coordinate frame f_j by only two view transformations $T_0^j \bullet T_0^i$. This topology reduces the propagation of registration error between any two views $\mathbf{V}_i$ and $\mathbf{V}_j$ by decreasing the number of transforms that must be chained together.

The star topology simultaneously "minimizes" the distance between any two nodes [7]. By using this topology (Fig. 9b) at most only two pairwise transformations are needed to project data from one viewpoint's coordinate frame to any other viewpoint's coordinate frame. Bergevin *et al*. [7] also show that the calculation of each transformation cannot be decoupled and done sequentially. View $\mathbf{V}_i$'s data may overlap more than one neighboring view. Therefore, the transformation T_i^0 taking points from $\mathbf{V}_i$ and the object coordinate frame not only effects the registration error between $\mathbf{V}_i$ and $\mathbf{V}_j$ but also any other view whose data overlap view $\mathbf{V}_i$. Finding the "best transformation" between any particular view $\mathbf{V}_i$ and the common object coordinate system uses all the correspondences between all the views simultaneously. This will not only minimize the registration error between view $\mathbf{V}_i$ and $\mathbf{V}_j$, but also minimize the registration error between view $\mathbf{V}_i$ and any other overlapping view $\mathbf{V}_k$, where $k \neq j$.

It is possible (due to self-shadowing and sensing errors) that even within the region of overlap there will be missing or unreliable data. With this in mind, Bergevin *et al*. [7] built on Chen and Medioni's work [23] by adding an additional heuristic designed to discard unreliable control points used in the registration process. The heuristic tests the control points and their corresponding tangent planes in the overlapping views for consistency in the orientation of the tangent plane and the normal at the control point. Normally, if there is only a small error in the registration, the normal of the control point and the normal to the tangent plane of an overlapping view should be nearly parallel, but due to larger than normal registration errors and errors in the scanning process, the difference can become quite large. This is a strong indication that the control point should not be used to refine the estimate of the registration between the views.

Neugebauer [80] used a graph topology to represent the relationships between views in the multiple-registration problem. The nodes of the network are the set of 3D data in each view and the links are the transformation matrices needed to map the data from one view to the coordinate system of a neighboring view. The goal is to obtain a well-balanced network where the registration error is similar for all pairs of views (Fig. 9b). A "central" coordinate system is chosen and initial transformation matrices are found to align each view with respect to this central coordinate system. These transformation matrices are then incrementally refined (using the Levenberg–Marquardt method) by minimizing the distance between points in one view and all corresponding points in the other views. This nonlinear optimization is guaranteed to reduce the registration error between the views. A statistical termination criterion is used based on an assumption that the residuals between views are zero-mean Gaussian random variables. To speed up the refinement process, only a few points from each view are used in the registration computation initially; as the refinement process continues, more and more points from the scene are incorporated into the estimate. This *resolution hierarchy* was shown to speed up the process of registering the views.

Pulli [90] documented a multiview-registration technique that begins with pairwise registrations, with a subsequent global alignment step. The latter step attempts to balance

registration errors across all views. Of note in this technique is its ability to handle a large data set size (i.e., a large number of scans and/or very dense scans), and a careful study of alternative alignment and optimization processes.

Alternative coordinate systems have proven useful in some registration contexts. Chen and Medioni [23] found it useful to convert all view data into spherical coordinates. With all the view data referenced to a single spherical coordinate system, they found it easier to combine the overlapping view data to a single-object model. In regions of overlap the corresponding points between views can be determined using the elevation and azimuth angular coordinates of points in the overlapping regions. These corresponding points are used to refine each view's transformation T_i^0 to the object's coordinate system. The resulting registered data were then transformed back to Cartesian coordinates.

Shum *et al*. [99] used a *resampling mesh* to represent each view and determine the transformation between them. The goal of the approach is to transform all the local coordinates to a global coordinate system that will represent the complete model, thus yielding the transformations for each view to the global coordinate system (Fig. 9b). The solution is found by generalizing a principal component analysis problem to a weighted least-squares problem. The approach shows noticeable improvements over sequential registration of the views. One drawback of this method is the use of resampling, which may cause some of the fine detail of the object to be lost.

In both Besl and McKay's [10] and Chen and Medioni's [23] work the process of finding the points or tangent surfaces that match in the error minimization process can be quite expensive. In ICP linear search is often employed while Chen and Medioni use a method similar to Newton's root-finding technique. To reduce the search complexity Blais and Levine [14] calibrate the sensor so camera parameters are known. With the camera parameters available, 3D coordinates can be accurately projected into the sensor's image plane. This allows range pixels from view $\mathbf{V}_i$ of an object to be quickly matched with range pixels of another view $\mathbf{V}_j$ by projecting pixels from the ith view into the image plane of view $\mathbf{V}_j$. This technique uses an initial estimate of the rigid transformation between the views, then refines it using corresponding points found by projecting pixels from one view into another view's image planes. Another difference between the work of Blais and Levine [14] and other multiple-view registration techniques is their use of simulated annealing to minimize the registration error instead of least squares of MVE estimators.

Eggert *et al*. [33] extended the traditional pairwise ICP algorithm [10] to solve the simultaneous registration of multiple views to achieve a global optimal registration for all views. In this new system all the data are transformed using coarse registration techniques into a single coordinate system. The data from each view are first smoothed using a median filter to remove outliers, then smoothed again using a Gaussian filter before estimating the surface normal at every point. The Gaussian-filtered data are only used for surface normal estimation and the median-filtered data are used in all other portions of their algorithm. At each iteration of their algorithm, point position and surface normal orientation are used to establish correspondence between points. During correspondence all the view data are searched to find the best match. Since this can be a lot of data they used hierarchical search along with space carving methods (*k-D* trees) to speed up the identification of the closest points. The refinement of the registration of the view data is accomplished by simulating a damped spring mass mechanical system. The corresponding points are linked with a hypothetical spring and incremental motions are calculated based on the forces and torques produced by the network of springs connecting the data sets together. Since the calculation of

point correspondence is still expensive, the force-based registration is allowed to iterate until convergence before new point correspondences are identified. Convergence is identified when the motion of the view data is sufficiently small. A drawback of this approach is that it requires that every portion of the surface be present in at least two views. The authors have presented some heuristics to allow them to work with and register data where this assumption is violated, but the authors still seek a more complete solution that does not require a heuristic based on thresholds.

4.2. View Integration

Once the registration process has been completed, the data contained in each view can then be transformed into a single-object coordinate system. Depending on the type of sensor that captured the data and the method of registration, the points on the surface can be in one of two basic forms. In the first case, all the points form a cloud where (at least initially) no connectivity information is known. In the second case, the data are structured via relationships (e.g, image–plane connectivity) known a priori in each view. The method of integrating the data depends on the form of the data (structured or unstructured) and the final representation required by the application.

4.2.1. Unstructured points. A polyhedral mesh constructed from the unstructured point cloud is useful for visualization and reasoning about the object's geometric properties. The potential difficulties with generating this approximation are as follows: recovering the correct genus of the object, dealing with missing data (undersampling of the surface), and search.

Since the data are initially unorganized, the task of simply identifying the set of points closest to a particular point in the cloud can be expensive computationally. There exist space-carving methods to speed up the search process (e.g., the *k-D* tree and uniform space partitioning) [39, 96]. With the emergence of connectivity information relating points in the cloud, some sense of the object's surface takes form. In particular, the orientation of the surface can be estimated more reliably from points within some neighborhood. Both Hoppe *et al*. [50] and Oblonsek and Guid [83] use an approximation of the surface normal to begin reasoning on the reconstruction of the surface shape.

The surface normal at a point in an unstructured cloud is easily calculated, but its orientation has a 180° ambiguity since the inside and outside of the object are not explicitly known. This information is important for recovering the correct genus of the object. To ensure that the correct genus is recovered, Hoppe *et al*. [50] used a graph optimization method to ensure neighboring points retain a consistent normal direction (neighboring normals point in the same general direction). Oblonsek and Guid [83] started with the point in the cloud with the largest z value and assumed that its normal was oriented in the $+z$ direction; subsequently, local consistency of the normal vector was enforced.

The normal direction attached to each point in the cloud gives a notion (at least locally) of which part of the tangent plane's half-space is inside or outside the object. This is used by Hoppe *et al*. [50] along with an isosurface to identify intersections of the object's surface with the cubes of the isosurface. Then by using a marching cube algorithm [70] they were able to recover a polygonal approximation of the object's surface.

Oblonsek and Guid [83] developed another method for forming a polygonal approximation to the surface represented in the point cloud. Their technique starts a seed point in the cloud, then grows the polygonal mesh using neighboring information and the surface

normal estimates. Their growing technique avoids the initial test of all the point's normal consistencies used by Hoppe *et al*. [50], while still allowing the system to recover objects of genus greater than 0.

4.2.2. Structured points. With structured input data (e.g., range images), the connectivity of the points in each view is already known. What remains is to integrate the data from separate views to complete a single representation of the object. The different methods for combining view data all address the following problems arising from the capture and registration of the data.

1. The sensor used to capture data produces errors in the measurements of the surface. This is especially present for some sensors along the silhouette of object in each view (where depth estimation techniques are least reliable).
2. The view data overlap in regions. The redundancy in the view data aids in finding the registration, but also ensures that accurate measurements of the object's surface exist in at least one view. These overlapping measurements must be integrated.
3. Self-occlusion and sensor errors result in holes in the data. The solutions to these problems, and to the general mesh integration problem, can be classified as mesh-based or volume-based.

In mesh-based techniques, the problem is to merge multiple polygonal meshes into a singular polygonal mesh that accurately represents the composite geometry. In the process of "stitching" together the views, seams can appear clearly showing the boundary between two separate views (Fig. 10). These seams result from several factors; registration error between views and sensor errors in the regions near the object silhouette. Both Soucy and Laurendeau [103] and Turk and Levoy [112] used an integration technique coupled with a geometric averaging technique to combine redundant data in the regions of overlap. They differ in the order in which they perform the operations. Soucy and Laurendeau [103] combine the redundant information from different views first to form a point set that is then stitched together. Turk and Levoy [112] first stitch together the meshes (removing redundant polygons in the overlapping regions) to form the final mesh, then use the vertices of the

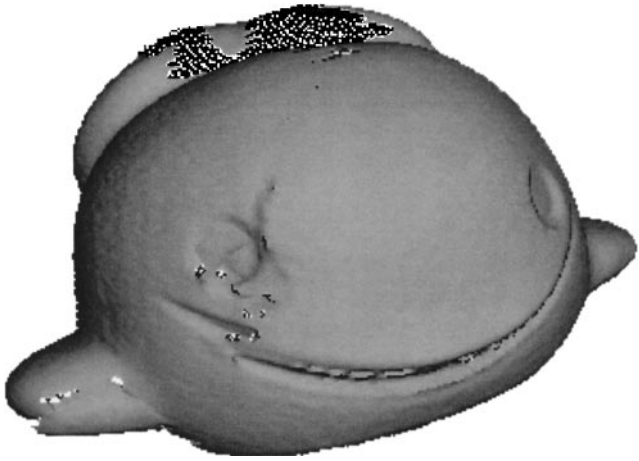

FIG. 10. A seam resulting from a registration error.

removed polygons to form a consensus of the surface shape in the regions of redundancy by moving the vertices of the final mesh. Both methods result in a local smoothing of the data in the regions of redundancy. Furthermore, neither method deals with holes in the data resulting from errors in the sensor measurement or due to the sensor's geometry. To fix the latter problem in their techniques, the holes were fixed by an operator that formed polygons in these regions, filling in the holes.

Soucy and Laurendeau [102] address more practical issues of surface integration for a computer-aided graphic design program. In these applications a designer will most likely take a series of views of the object to build an initial model of the object being modeled. Then because of self-occlusion and sensor errors the designer often must take scans of additional views of the object to fill in large holes or recapture fine detail in the object. In this case repeating the integration of all the views simultaneously for every additional view may not be the best solution especially for an interactive model-building program. Instead Soucy and Laurendeau have developed a dynamic integration algorithm that adds additional data to a model by modifying only the necessary data structures. The new algorithm maintains the advantages of simultaneous integration techniques while maintaining the ability to sequentially integrate new data to a model.

The volume-based set of techniques limit the reconstructions to objects with a genus of 0. Under this assumption, a representation can be formed without worrying about the difference between samples resulting from an actual hole in the object and a hole resulting from sensor errors.

Under the genus zero assumption, a deformable surface can be used conform to the object's data points. These deformable surfaces can be thought of as a ballon being inserted inside the object then inflated until the surface of the balloon conforms to the data (Chen and Medioni [23]). Alternatively, a filled balloon can be placed around the object then slowly the air can be released until the balloon forms a mold of the data (Shum *et al*. [99]). These techniques solve several problems related to the registered data. First the deformable model implicitly handles redundancy in the data, since the deformable model contains its own vertices, which are moved based on an energy minimization between the model and the data. Second, the deformable model will fill in the holes in the data by stretching the deformable model's surface between the boundaries of the hole. However the fine details are often lost due to the smoothing effect of the energy minimization and the fixed set of vertices in the deformable mesh. Because of this, Chen and Medioni [23] added adaptive mesh refinement of the deformable surface, which allows the model to adaptively subdivide in order to expand into regions of fine detail that may have been lost otherwise.

A novel volume-based technique developed by Reed *et al*. [94, 93] used a constructive solid geometry (CSG) technique to form a solid model of the object from multiple registered range views. Each view forms a solid by sweeping the range data from each object away from the scanner, filling in the space self-occluded by the object. The surfaces of this new object are labeled as being either visible (part of the sensed data) or occluded by the object view. This labeling can be used for view planning since it labels portions of the viewspace that have not been covered by the previous view. For each view, solid objects are formed using similar sweeping operations. Once all of the visible portions of the object have been covered by one or more views, the view solid models are intersected to form a singular solid object model.

4.3. Mesh Refinement and Optimization

Due to the discrete nature of the data being used to model the object, the result will only be an approximation. The accuracy of that approximation depends on the ability of the sensor to capture data and ability of the system to register multiple data sets. With the availability of high-resolution range scanners and better methods of registration, the quality of the reconstructed model can be quite good. Still, there may exist a need to refine the model based on application requirements [46].

In graphics applications the important properties of a refined model are that it occupy roughly the same shape, that its reconstructed surface normals be close to those of the original model (important for shading), and finally that the object's color texture can be mapped to the reconstructed model. All these factors improve the model's ability to yield synthetic images that appear very similar to the true scene and the initial unrefined model.

The neighboring vertices in a good-quality polygonal mesh should be geodesically the same distance apart (Welch and Witkin [115]). For highly curved areas on an object, two vertices might be close geometrically, but very far apart geodesically (due to surface curvature changes). In these regions, the mesh representing the surface should be fine, while in regions where the surface is largely planar the density of the mesh can be coarse. This adaptive mesh density results in very good approximation of the object's surface normal and curvature, and is a key goal of mesh refinement procedures.

The speed of the rendering depends directly on the number of polygonal faces in the mesh. Because of this, *mesh optimization* strategies have been studied to reduce the number of polygons needed to represent the model for the object [46, 111, 49, 47, 69, 26, 41, 48, 104, 101], while still maintaining a good approximation to the surfaces. Figure 11 shows an example of using mesh optimization strategies to reduce the number of polygons in the representation of a toy whale.

In the process of refining a model, care must be taken to avoid simplifications that yield a topological change (e.g., creation of holes and surfaces smoothed to a line). This can happen when vertices are removed from the mesh and the mesh is then repolygonized. To take care of this problem, most systems use heuristics to verify that the model's topology does not change [111] during the refinement process.

During decimation, the process of removal of vertices, edges, and faces can result in a smoothing and/or shrinking of the object's surface. Care must be taken while refining the object's polygonal representation to avoid smoothing out the original model's sharp discontinuities and therefore shrinking of the object. Automated techniques tend to utilize a penalty function that measures the error incurred by iteratively refining the number of polygons in the mesh [46, 111, 49, 47, 69, 26, 41, 48, 104]. These functions guide the choices of which entities (vertices, edges, or polygons) should be removed and where new entities should be placed.

In addition to decimation, another approach to reducing the amount of space needed to store a model is to change the representation of the model to something that is less space consuming than a dense polygonal mesh. Krishnamurthy and Levoy [65] studied the conversion of a dense polygon mesh to a B-spline surface patch representation for objects. An appropriately designed B-spline patch representation uses less computer storage space than a polygonal mesh that represents the surface with the same accuracy. In addition to being more memory efficient, the B-spline is differentiable and has been extensively used in the CAD/CAM industry for representation and design [35].

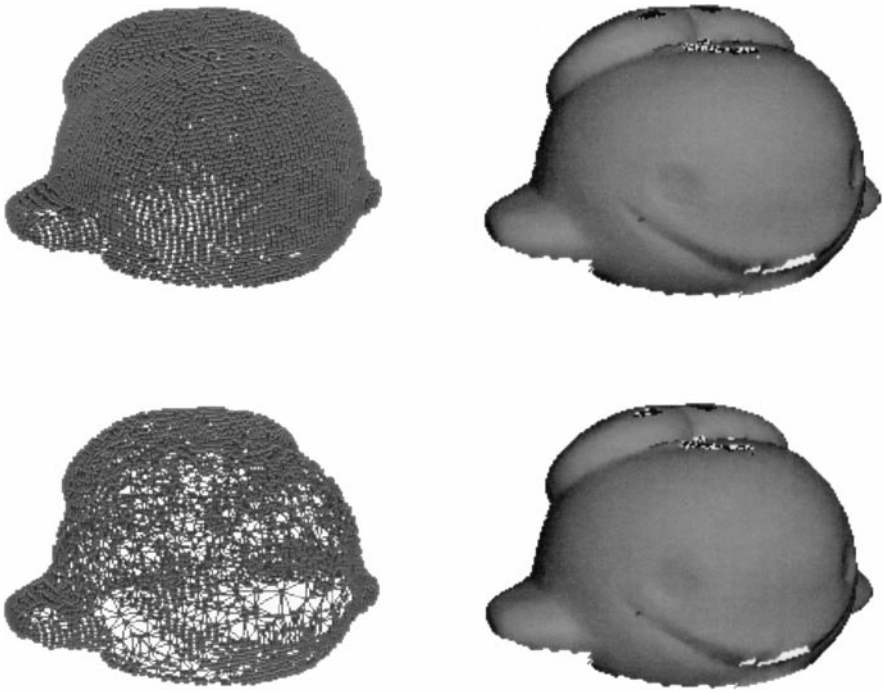

FIG. 11. In the top pair of images the whale mesh contains 6844 vertices. The bottom set of images shows a the result of reducing the mesh to contain only 3042 vertices. The right image in both pairs is a shaded version of the mesh to the left. The images were generated using Minolta's Vivid software package.

For the area of computer vision the criteria for a good representation of an object model may be different from the criteria employed in computer graphics. For example, Johnson and Hebert [55, 58] required that the models of the objects in their database be represented in polygonal mesh format with the density of the mesh as regular as possible. The regular mesh ensures that their "spin" image features (used to encode and match the shape of each object) approximates the true range image features scanned from the real object's surface. The use of spin images for recognition is discussed at length in Section 5.3.

Another example is the identification of curvature extrema from polygonal mesh representations of surfaces. Curvature is often used in computer vision as a descriptive feature because of its detectability in a wide variety of imagery. Campbell and Flynn [19] studied some of the issues in identifying contours of curvature extrema on polygonal mesh surfaces. They found that identification of the curvature extrema on an object surface depended on the properties of the polygonal mesh used to represent it. Polygonal mesh representations with vertices guaranteed to be within an error tolerance of the true surface typically will contain polygons with varying surface area, depending on the magnitude of the the surface curvature. In regions where the surface is rapidly changing shape the polygonal mesh model must use small polygons to ensure the error between true surface and polygonal model is within tolerance. This difference in area can then be used to aid in the identification of the curvature extrema on the mesh representation of the model.

Another concern is identifying ridges of curvature extrema. Campbell and Flynn [19] developed a curvature ridge/valley detector that used hysteresis and nonmaximum suppression to build contours of curvature extrema. They found that following the true contour was

often difficult since the edges of the polygonal mesh representation can be a poor polygonal approximation to the contour of curvature extrema. This results from poor approximation of the ridges of the object surface by the edges in the polygonal mesh. Often the edges of the mesh cut through the true contour instead of following it. This haphazard placement of edges in the polygonal mesh causes trouble in reconstructing contours from the representation.

The refinement of the model in large part depends on the application and the desired accuracy of the representation. A resampling of the data is often performed to improve the ability to reconstruct approximations to the surface normal and curvature on the model [19] and to reduce the redundancy in the representation. The result is a model that more accurately represents the object and all of its properties while still being efficient in time and space.

5. FREE-FORM OBJECT RECOGNITION SYSTEMS

The preceding sections have surveyed the elements and techniques for representing free-form 3D objects, with a focus on mesh representations and their construction from range images. This section discusses several approaches to the recognition of such objects in intensity or range imagery. Additional surveys of this area (or of object recognition in general) include Besl and Jain [9], Brady *et al*. [15], Flynn and Jain [37], and Pope [89]. Our goal is to survey those systems that are relatively recent, are designed to recognize either single or multiple objects in the presence of occlusion, and do not restrict the 3D shape of objects to be identified to simple shapes such as planes or cylinders. While some attention is given to recognition in intensity imagery to provide context, the focus here is techniques employing 3D (range) imagery.

5.1. Appearance-based Recognition

Appearance-based approaches have become a popular method for object identification and pose location in intensity images, and has also been used to recognize free-form objects in range data, as noted below. This popularity is in part due to these methods' ability to handle the effects of shape, pose, reflectance, and illumination variations for large databases of general objects. An appearance-based recognition system encodes individual object views as points in one or more multidimensional spaces. The bases for these spaces are obtained from a statistical analysis of the ensemble of training images. Recognition of an unknown view is typically performed by projecting that view into the space(s) along the stored basis vectors and finding the nearest projected view of a training image.

The literature contains many variations on this simple and powerful, basic idea. Early work in the area of appearance-based recognition included a study of efficient representation of pictures of faces. Kirby and Sirovich [63] used principal component analysis to determine the best coordinate system (in terms of compression) for their training data. The basis vectors for the new coordinate system were termed *eigenpictures*. In later publications [114, 75] they are sometimes given equivalent terms, *eigenfaces* or *eigenimages*, depending on the application. The term *eigenfaces* is used when the training set is limited to face data because certain basis vectors, when viewed, have a facelike appearance.

The appearances of objects are encoded using principal component analysis on a set of training image data

$$\mathbf{X} = \{\vec{\mathbf{x}}_1, \ldots \vec{\mathbf{x}}_M\}.$$

Each training image $\vec{\mathbf{x}}_i$ can be considered a column vector obtained from the original image $[x_{i;jk}]$ by unraveling

$$\vec{\mathbf{x}}_i = [x_{i1}, \ldots x_{iN}]^{\mathrm{T}},$$

where N is the number of pixels in the image. Each image is typically normalized to unit energy $|\vec{\mathbf{x}}_i| = 1$. For Lambertian objects the normalization factors out any variation in global illumination that might occurr between trials. To accurately represent each object's appearance, the object must be viewed in a set of poses and under a set of illumination conditions that captures those poses and conditions expected to be present in the recognition environment.

Principal component analysis is used to determine a set of orthogonal basis vectors for the space spanned by the training data $\mathbf{X}$. To do this the average

$$\vec{\mathbf{c}} = \frac{1}{M} \sum_{i=1}^{M} \vec{\mathbf{x}}_i$$

is subtracted from every image $\vec{\mathbf{x}}_i$ in $\mathbf{X}$, yielding a centered set $\mathbf{X}_c$. $\mathbf{X}_c$ is then used to form an $N \times N$ covariance matrix:

$$\mathbf{Q} = \mathbf{X}_c \mathbf{X}_c^{\mathrm{T}}.$$

Then the eigenvectors and corresponding eigenvalues $\{(\vec{\mathbf{e}}_i, \lambda_i), i = 1 \ldots N\}$ for $\mathbf{Q}$ are determined. The set of N eigenvectors is an orthogonal basis for the space spanned by $\mathbf{X}$. This principal components representation can be used to approximate the training data in a low-dimensional subspace that captures the gross appearance of the objects. This enables appearance-based approaches to characterize the objects with the eigenvectors corresponding with the largest eigenvalues. Murase and Nayar [75] found that a subspace of 20 dimensions or less was sufficient to capture object appearance for pose estimation and object recognition purposes.

Let $\{\vec{\mathbf{e}}_1, \ldots, \vec{\mathbf{e}}_k\}$ be the set of eigenvectors corresponding with the k largest eigenvalues. The centered vectors $(\vec{\mathbf{x}}_i - \vec{\mathbf{c}})$ in the training data are projected along $\{\vec{\mathbf{e}}_1, \ldots, \vec{\mathbf{e}}_k\}$ to form k-dimensional prototypes $\vec{\mathbf{g}}_i$. An unknown image $\vec{\mathbf{s}}$ that maps close to one of these prototypes is similar in appearance to the corresponding training image $\vec{\mathbf{x}}_i$. This enables the system to identify object identity and pose.

An approximation to $\vec{\mathbf{x}}_j$ can be constructed using the basis vectors $\vec{\mathbf{e}}_i$ and the projected point $\vec{\mathbf{g}}_j$ in the eigenspace. The reconstructed image is given by

$$\tilde{\mathbf{x}}_j = (\vec{\mathbf{e}}_1, \ldots, \vec{\mathbf{e}}_k)\vec{\mathbf{g}}_j + \vec{\mathbf{c}},$$

and is only an approximation to $\vec{\mathbf{x}}_j$ since only k eigenvectors are used instead of $\min(N, M)$ (the rank of the covariance matrix $\mathbf{Q}$), but the gross details of the object's appearance are typically preserved.

Turk and Pentland [114, 113] noted that in using appearance-based recognition strategy a small set of eigenpictures could be used to describe and reconstruct faces from a large segment of the population. Their ideas were tested on a database of 16 subjects whose faces were captured under varying illumination, pose, and scale (2500 images in all). They found that the system had a higher sensitivity to changes in the size of the subject

face in the digitized images than to changes in illumination and pose. They showed that appearance-based methods could be used to recognize complex sculpted surfaces (faces) in under 1 s.

Murase and Nayar [75] developed the use of appearance-based recognition and pose recovery for general objects. Using two eigenspaces they were able to recognize and locate pose for objects under varying lighting conditions. The universal eigenspace $\mathcal{U}$ is formed from all training images $\mathbf{X}$ and is used to determine the identity of an object in a scene image, while an object eigenspace $\mathcal{O}_j$ is formed for each model j from all training images $\mathbf{X}_i$ containing that model, and used to determine the object's pose and the illumination conditions. For each object a bivariate manifold is created, allowing interpolation between discrete points $\vec{\mathbf{g}}_i$ in the eigenspaces. The manifolds were constructed using a cubic-spline interpolation algorithm. The two parameters of the manifold were the rotation of the imaging turntable and the illumination direction. In general, three parameters are required to represent the rotational pose of an object in 3D and an additional two dimensions are needed to describe varying illumination positions (for a single nondirectional light source). Thus the practicality of this sort of system depends on the number of pose and illumination degrees of freedom as well as the complexity of the object shape and appearance under pose and illumination variations.

To identify an object in an image $\vec{\mathbf{s}}$, it is first projected

$$\vec{\mathbf{g}} = \begin{pmatrix} \vec{\mathbf{e}}_1^T \\ \vdots \\ \vec{\mathbf{e}}_k^T \end{pmatrix} (\vec{\mathbf{s}} - \vec{\mathbf{c}})$$

into the universal eigenspace, then matched with the closest object's manifold. Naively, matching can be done by uniformly sampling the manifolds, but this is both inefficient in memory and time. Nene and Nayar [78, 79] developed a structured binary search technique to quickly search through a multidimensional eigenspace for the best match.

Other notable work in the appearance-based recognition of objects in intensity scenes includes the following.

- Turk and Pentland [113] discussed the utility of using trained neural networks to match $\vec{\mathbf{g}}$ with a person's identity.
- Mukherjee and Nayar [72] experimented with radial basis function (RBF) neural networks to match $\vec{\mathbf{g}}$ with a point on an object's manifold.
- Murase and Nayar [74] followed their earlier work on appearance-based matching of object with a study of illumination. The goal of the study was to determine illumination parameters to maximize the difference in appearance between objects. The study produced graphs of minimum distance between object curves in eigenspace as a function of light source direction, recognition rate as a function of SNR, and recognition rate as a function of the amount of segmentation error. In all cases the experimentally determined optimal source was found to have a higher recognition rate in the presence of noise and segmentation errors.
- Mundy *et al.* [73] compared Murase and Nayar's [75] appearance-based approach with two geometric model-based recognition approaches. The study concluded that the appearance-based system had the highest recognition rate, but it also had the highest number of false positives. This was due to the lack of a verification procedure, and to the sensitivity

of current appearance-based methods to occlusion, outliers, and segmentation errors. The geometric model-based approaches, being structured around the hypothesize-and-test architecture, had a smaller number of false positives.

- Black and Jepson [13] showed that the projection $\vec{\mathbf{g}} = [\vec{\mathbf{e}}_1, \ldots \vec{\mathbf{e}}_k]^{\mathrm{T}}(\vec{\mathbf{x}} - \vec{\mathbf{c}})$ results in a least-squares estimate when used for reconstruction (i.e., the calculation minimizes the squared error between image $\vec{\mathbf{x}}$ and its reconstruction $\vec{\mathbf{g}}$). To compensate for the well-known sensitivity of least-squares techniques to outliers and gross errors, Black and Jepson used robust statistics in the calculation of the projection $\vec{\mathbf{g}}$. The addition of robust estimation improved the appearance-based method's ability to handle input images for which good segmentations were not available. A notable application was the development of a system to track and recognize hand gestures.

- A major problem with the appearance-based approaches is their lack of ability to handle more than one object in the scene with the possibility of occlusion. Huang *et al*. [51] proposed segmenting the input images into parts and using the appearance of the parts and their relationships to identify objects in the scene as well as their pose. Later work by Camps *et al*. [21], following Huang *et al*. [51], enhanced and extended the earlier work, yielding a recognition system that includes hierarchical databases and Bayesian matching.

- One drawback of an appearance-based part representation is the input image must be segmented at runtime before recognition can occur. This limits the class of objects that can be recognized to objects who can be segmented reliably. Most free-form objects do not lend themselves to easy repeatable segmentations. Ohba and Ikeuchi [84] and Krumm [66] avoided this problem by creating an appearance-based technique using local windows of the object's appearance. Ohba and Ikeuchi [84] were able to handle translation and occlusion of an object using *eigenwindows*. The eigenwindows encode information about an object's appearance for only a small section of its view. Measures of *detectability*, *uniqueness*, and *reliability* were developed for the eigenwindows. These measures are used to omit eigenwindows from the training set if they are hard to detect (detectability), have poor saliency (uniqueness), or are sensitive to noise(reliability). Using the local eigenwindows they were able to identify multiple objects in cluttered scenes. Krumm [66] independently and roughly simultaneously devised a eigenwindow approach similar to that of Ohba and Ikeuchi [84]. The differences between these two approaches deal with the model construction and recognition procedure. Krumm employs an interest operator (points and corners) to identify possible nonoverlapping eigenwindows.

- Edwards and Murase [32] addressed the occlusion problem inherent in appearance-based methods using a mask to block out part of the basis eigenimages $\vec{\mathbf{e}}_i$ and the input image $\vec{\mathbf{s}}_i$. The masks were used with a resolution hierarchy to search for an initial mask and object identity at low resolution, then adaptively refine them through higher resolutions.

- Leonardis and Bischof [67] handled occlusion by randomly selecting image points from the scene and their corresponding points in the basis eigenvectors $[\vec{\mathbf{e}}_1, \ldots, \vec{\mathbf{e}}_k]$. Their method uses a hypothesize-and-test paradigm, where a hypothesis is a set of image point locations (initially randomly generated) and the eigenspace prototype $\vec{\mathbf{g}}$. This method shows the ability to reconstruct unseen portions of the objects in the scene. Bischof and Leonardis [12] extended this earlier work to handle scaling and translation of objects in the scene.

- Rao [92] applied the adaptive learning of eigenspace basis vectors $[\vec{\mathbf{e}}_1, \ldots, \vec{\mathbf{e}}_k]$ in appearance-based methods. The dynamic appearance-based approach is used to predict spatial and temporal changes in the appearance of a sequence of images. The prediction modifies the eigenvectors by minimizing an optimization function based on minimum

description length (MDL) principles. MDL is used because it balances the desire to learn the input data without memorizing its details. This is important when the system needs to be flexible and not overspecialized to the training data.

- Appearance-based recognition has also been applied to range or depth imagery by Campbell and Flynn [20]. The $2\frac{1}{2}$D depth data capture the shape appearance of a view. The principal component analysis of this data results in a set of characteristic shapes. These *eigenshapes* are used in much the same way as *eigenpictures* or *eigenfaces* to form a low-dimensional subspace that is useful for matching and pose recovery. An advantage to using range data in place of albedo is that in most cases the illumination conditions at the time of the scan does not affect the measurements of the depth data in the range image. Traditionally appearance-based recognition systems trained on images captured by rotating the object on a turntable and varying the elevation of the light with respect to the turntable. This resulted in two degrees of pose freedom, the first for the rotation of the turntable, and the second for the light position that could be recovered by the appearance-based recognition system. Since illumination was not a factor in the shape appearance, the authors addressed the recovery of 3D rotational pose.

The training of the system consisted of uniformly sampling the view sphere by a pseudo-regular tessellation of the sphere. Each vertex of the discrete view sphere defines a viewpoint for which a set of training images are taken. The training images were then used to build, along with Nene and Nayar's [79] efficient search technique in high-dimensional spaces, an efficient object recognition system for free-form shapes. This system was tested on two databases; the first contains 20 free-form objects and the second contains 54 mechanical parts (most of the surfaces here could be described by the natural quadrics). Both databases were tested to identify the influence that density of pose sampling, dimension of the matching subspace, and size of training images had on the probability that the correct object would be detected. The authors found that image size was not nearly as important as the number of training images taken for each object and the dimension of the subspace. Campbell and Flynn also noted that an appearance subspace of 20 dimensions was sufficient for accurate object recognition. Their system was able to identify the objects using shape with 91% accuracy in about a second on a Pentium class PC.

Table 2 summarizes the various appearance-based recognition techniques surveyed above. For each technique, the table indicates whether the technique can handle multiple-object scenes with occlusion and changes in the size of the objects in the database. It also reports the largest documented database size and the recognition rate obtained $\frac{Nc}{Ntrials}$, where Nc was the number times the object's were correctly identified and Ntrials was the total number of trials.[2] In some of the papers surveyed the number of objects or the recognition rate was not reported and are marked in the table as NR. The entry for the system of Black and Jepson [13] reports N/A for database size and rate because it was designed to track objects using recognition and multiple-object databases were not explored.

5.2. Recognition from 2D Silhouettes

In addition to appearance-based techniques, object silhouettes have been used to characterize free-form objects. In a controlled environment an object's silhouette can be quite

[2] In the experiments of Nayar and co-workers [75, 77], tests using a 100-object database reported 100% correct recognition for a database subset of 20 objects that do not contain self-similar viewpoints.

TABLE 2
Summary of Appearance-based Recognition Techniques

Technique	Occlusion	Scale changes	Largest database	Recognition rate
Kirby and Sirovich [63]	No	Yes	NR	NR
Turk and Pentland [113]	No	No	16	1.0
Murase and Nayar [75], Nayar *et al*. [77]	No	Yes	100	1.0
Black and Jepson [13]	No	No	N/A	N/A
Camps *et al*. [21]	Yes	No	24	1.0
Ohba and Ikeuchi [84]	Yes	No	NR	NR
Krumm [66]	Yes	No	2	0.8
Edwards and Murase [32]	Yes	Yes	6	0.89
Bischof and Leonardis [12]	Yes	Yes	20	NR
Rao [92]	Yes	No	NR	NR
Campbell and Flynn [20]	No	Yes	54	0.91

useful to determine an object's identity and pose. This subsection reviews a few representative techniques employing free-form contours in intensity imagery for recognition.

Mokhtarian [71] developed a complete object recognition system based on closed object silhouettes. The system is designed to recognize free-form objects that have only a few stable views in an environment where only one object will be present. To do this, a light box is used to illuminate the object and make the boundary between background and object easier to detect, and objects are isolated by simple thresholding. Boundary curves (extracted by contour following) are then represent by calculating its curvature scale space (CSS) representation. The matching of CSS curves is done based on the location of the maxima of the curvature zero-crossings. For convex boundary curves the CSS representation is smoothed until only four maxima points are remaining, while concave curves utilize all maxima obtained at a given scale. During recognition the aspect ratio of the object's silhouette is used to prefilter possible scene/model matches before the silhouettes are matched. This technique provides a fast way of matching the coarse features of a scene silhouette with an object's silhouette. The best silhouette matches are then verified by registering the two curves and measuring the error. The system correctly identified all 19 objects in its database.

Ponce and Kriegman [88] used contours of discontinuous surface normal and occluding contours to recognize 3D objects. Parametric surface models of the objects yield candidate model contours. Using elimination theory, implicit equations of the parametric patches are found and the intersections between the patches are used to construct an implicit equation of the contour. A weak perspective projection model is used to correlate 3D contours with 2D contours found in the intensity images. Image contours are manually pruned and grouped into clusters corresponding to a single object. The system was only used to model surfaces of revolution, but could handle multiple object scenes with moderate amounts of occlusion. Their results showed quick recognition times ($\approx$30s) with good location accuracy (average error between 0.4 and 1.4 pixels).

Joshi *et al*. [61, 62] used HOT (high-order tangent) curves to model a smooth object for recognition. These curves can be identified from points on an object's silhouette in an intensity image. Sequences of these points can be tracked to recover the 3D curve. The angles between tangent lines and ratio of distances between points on contour are useful to

form scale and pose independent features of the object. This in turn can be used to index a recognition table. Their experiments tested the system on four objects (a squash, a pear, a banana, and a duck decoy) and showed that the HOT curve representation was able to recover the objects identity and pose for every test, but only after a verification step pruned out false matches.

Internal edges have also been used to help recognize free-form objects in intensity images. Chen and Stockman [22] used both silhouette and internal edges to recover pose and identity of 20 free-form object models from intensity imagery. The silhouette curve was first used with a invariant feature indexing system to build candidate model and pose hypotheses. A part-based contour representation was incorporated to tolerate occlusion. The invariant attributes of these curve segments were then used to index into a hash table. This results in matches between possible model parts and observed parts. These matches are then grouped into consistent hypotheses based on model identity (associated with each part) and a rough pose estimate. The hypotheses are then verified by matching model edge maps to the observed edge maps. The verification step also produces a refined estimate of the object's pose. During verification both the silhouette and internal edges are used. The inclusion of the internal edges improved the performance of the verification step in rejecting false hypotheses of object identity and pose.

5.3. Free-Form Object Recognition in Range Data

Besl [11] reviewed the difficulties in matching free-form objects in range data using point, curve, and surface features. The computational complexity of such matching procedures can quickly become prohibitive. For example, brute-force matching of 3D point sets was shown to have exponential computational complexity. Because of this, Besl and McKay [10], Stein and Medioni [105], Johnson and Hebert [57, 54, 56], and Chua and Jarvis [25] all have developed techniques to reduce the amount of computation required. These methods often group corresponding model and scene point pairs into sets that can be used to determine object pose and verify the existence of the model in the scene.

Range images allow the computer vision practitioner more complete information about object geometry. Their ability to capture 3D points on the viewable surfaces can be exploited to compare geometric relations between range images and training data or 3D models. While their utility in construction of 3D models has been a topic of relatively recent interest (as surveyed above), the recognition of objects in depth maps has a longer history and systems developed for this purpose have demonstrated a variety of approaches. The purpose of this section is to describe recent systems that were developed for free-form object recognition using range images.

A technique formulated by Nevatia and Binford [81] used symbolic descriptions derived from a generalized cone part segmentation of range images to recognize free-form articulated objects (doll, horse, snake, glove, and a ring) in the presence of occlusion. The parts (generalized cones) are used to form a symbolic description of the scene built up using part properties (e.g., axis length, average cross-section width), connectivity relations (e.g., number of parts connected), and global properties (e.g., number of parts, number of elongated parts). In an effort to eliminate some of the objects, distinguished parts are employed to index into the database of possible objects. Distinguished parts are generalized cones that are much wider than other parts in the database. These parts are more likely to be segmented properly in the presence of occlusion. The scene and model descriptions are then

compared, starting with matching distinguished parts. These matches are then grown by matching other distinguished parts whose properties and connections are consistent. This is equivalent to matching two graph descriptions of the objects. The resulting scene graph of connected parts is allowed to be smaller than the model graph because of the presence of occlusion and segmentation errors. The quality of the graph match is then evaluated by the quality of the individual part matches. This representation works well for objects who have very different symbolic descriptions.

Raja and Jain [91] developed a system for fitting and classifying deformable superquadrics (a subset of the set of all free-form objects). They fit a deformable superquadric to range data where the deformations considered were tapering and bending. The parameters that define the fit superquadric are used to classify the shape into one of 12 different geon classes. The geon classes are used to discriminate between "qualitatively different" shapes. Qualitative differences include straight vs curved axes, straight vs curved cross section, and increasing, decreasing, or increasing-then-decreasing cross-sectional area along the primary axis of the superquadric. Superquadrics recovered from noisy or rough range images caused problems with classification of the superquadric with the correct geon class. With real range images their classification method correctly identified the geon class 77% of the time and with synthetic imagery the correct identification rate was 87%.

Surface curvatures has also been used to describe an object's surface. For a surface the curvature at a point is characterized by finding the directions in which the surface normal changes the most and the least rapidly. These *principal curvatures* $\mathcal{K}_1$ and $\mathcal{K}_2$ respectively encode the rate of change of surface orientation in the two extremal directions. Typically the principal curvatures $\mathcal{K}_1$ and $\mathcal{K}_2$ are used to characterize and encode discriminatory information about an object.

Thirion [110] used surface curvature to define a global representation of free-form objects. The method uses curvature extrema on the surface to find critical points and contours. The extremal point/contours are defined as zero crossings between maxima and minima of Gaussian curvature ($\mathcal{K} = \mathcal{K}_1 * \mathcal{K}_2$). The representation has been used on real range data as well as synthetic models to find extremal meshes of the object. Thirion notes that the representation is still too complex to be practical for object identification or pose localization, because the description does not lend itself easily to a quick matching technique.

Surface curvature has also been used to classify local surface shape into a small set of representative shapes [9, 31]. Besl and Jain [9] used Gaussian curvature ($\mathcal{K} = \mathcal{K}_1 * \mathcal{K}_2$) and mean curvature ($\mathcal{H} = (\mathcal{K}_1 + \mathcal{K}_2)/2$ to classify local surface shape into eight basic categories. Dorai and Jain [31] extended this earlier work by defining two new curvature measures: the *shape index* ($\mathcal{S} = 1/2 - \frac{1}{\pi} \arctan \frac{\mathcal{K}_1 + \mathcal{K}_2}{\mathcal{K}_1 - \mathcal{K}_2}$) and *curvedness* ($\mathcal{R} = \sqrt{(\mathcal{K}_1^2 + \mathcal{K}_2^2/2)}$, where the shape index $\mathcal{S}$ now defines (classifies) the local shape into nine shape types (very similar to Besl and Jain's work [9]) and the curvedness measures the magnitude of the curvature change. Dorai and Jain [31] use these new measures along with a spectral extension of the shape measure to build a view-dependent representation of free-form objects. Their system (named COSMOS, for "curvedness-orientation-shape map on sphere") uses a histogram of shape index values to characterize the surface curvature of a view. The histogram bins store the amount of area on a view that lies within a range of shape index values. These histograms (called shape spectra) can be quickly matched using moments and are invariant to rotations about the optical axis. The system builds up an object database made up of many views of each object to be recognized. To reduce the complexity of the search, views of an object are grouped based on their similarity into clusters. Then for each cluster a prototype

shape spectral histogram is found using averaging. The process to match a scene shape spectra histogram with the database first matches the scene with each cluster's prototype. Then the top n clusters that match well with the scene are examined to find which views in the clusters best match the scene. The view that best matches the scene identifies the object and pose. The translation and rotation parameters have yet to be determined. The view clustering scheme for the recognition database reduces the amount of time it takes to produce a match. On average only 20% of the views had to be matched from a database with a total of 20 objects. This recognition scheme works well for single-object scenes not containing polyhedral objects.

In addition to curvature-based features, deformable polyhedral meshes have been used to encode the local surface shape. Pipitone and Adams [87] used a connected set of equilateral triangles (Fig. 12) to characterize an object shape by deforming it to the shape of the surface. The crease angle (Besl [8]) between pairs of triangles in the mat are used to build a pose invariant feature vector Θ that characterizes the local surface orientation change. The number of angles N_a resulting from the mat of triangles gives the order of the operator Θ, where

$$\Theta = \begin{pmatrix} \theta_1 \\ \vdots \\ \theta_{N_a} \end{pmatrix}.$$

The length of the sides of the equilateral triangles can be used to control the operator's ability to capture fine detail or reject noise in the surface shape of the object. To completely encode an object, the tripod operator is randomly placed on its surface enough times to ensure sufficient coverage of all the surfaces of the object. Recognition in this context was

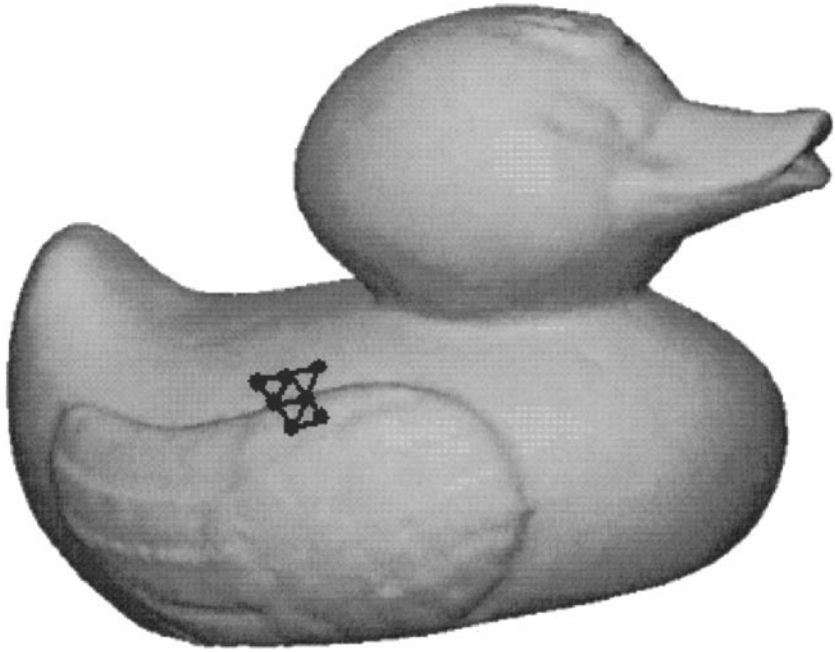

FIG. 12. Example of a fourth-order operator on range data sampled from a rubber duck.

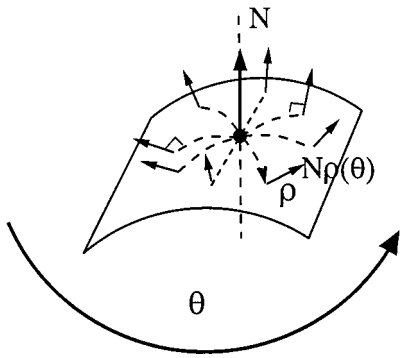

FIG. 13. The splash representation. The angular differences between the normals $N\rho(\theta)$ near the central point and the normal N at the central point are encoded.

performed using maximum a posteriori (MAP) estimation of the object identity given the observed tripod features. Nonparametric density estimates were employed in the MAP estimator. By using 10 tripod operators and picking the most likely object, a correct recognition rate of 92% was achieved for a four-object database.

Stein and Medioni [105] used changes in surface orientation to match local patches of surfaces. The local nature of the matching technique allowed them to find multiple free-form objects in a cluttered scene. To provide good matches between corresponding scene points and known model points they devised a novel method for measuring the difference between two relative surface normal distributions (Figs. 13, 14). For a given point P on a surface the normals a distance ρ away from P contain some structural information about the surface around P (Fig. 13). The distribution of all the normals $N\rho(\theta)$ on the surface around P are called a "splash," because its appearance can be strikingly similar to a splash in water. Then to encode relative information about the normals $N\rho(\theta)$ a spherical coordinate system is used (Fig. 14), where the angles $\phi(\theta)$ and $\psi(\theta)$ give the relative orientation of $N\rho(\theta)$ with respect to P's normal N and the $X(\theta)$ axis. $X(\theta)$ is perpendicular to N and lies in the plane containing P, N and the point ρ distance away from P and angle θ from where the encoding started. As θ is varied from 0 to 2π the values of $\phi(\theta)$ and $\psi(\theta)$ form a 3D curve

$$\mathbf{v}(\theta) = \begin{pmatrix} \phi(\theta) \\ \psi(\theta) \end{pmatrix}.$$

To allow for quicker matching techniques between pairs of curves $\mathbf{v}_i(\theta)$ and $\mathbf{v}_j(\theta)$ the

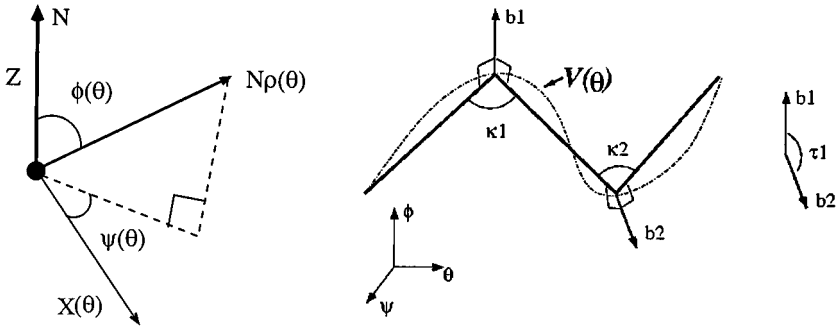

FIG. 14. Orientation coordinates for splash features.

curves are polygonized starting at the value of θ where $\phi(\theta)$ is maximum. This starting point is chosen to provide one method of ensuring some rotational invariance to encoding the curve. Then finally the polygonal curve is encoded into a representation called the *3D supersegment* (Fig. 14). The 3D supersegment stores the curvature angles between links κ_i and the torsion angles τ_j shown in Fig. 14 from the polygonization of the curve $\mathbf{v}(\theta)$.

For recognition, the best places to encode splash features are in the areas of high curvature. In these areas the variations between the normal at P and the normals $N\rho(\theta)$ give splashes a rich structural description of the local surface shape of the object.

Stein and Medioni [105] employed a "structural indexing" approach to matching. This recognition method is a variant on hashing where the indices to the hash table are related to structures formed from the features. In this case the code for the hash table is found from the 3D supersegment. The curvature κ_i and torsion angles τ_j between the segments of the polygonal curve along with the number of segments, the maximal orientation difference between point of interest and the contour max $\phi(\theta)$, and the radius ρ are used to index the hash table. This is termed structural indexing because the table is indexed by structural elements of the features. For a given interest point a entry is encoded for various radii and number of segments in the polygon approximation of the splash. At recognition time scene interest points are determined and used to index the hash table. The matches are used to generate hypotheses. These are first grouped by the model they correspond to. Then for each model a set of geometric-consistent hypotheses are formed. The geometric consistency heuristic checks to see if pairs of point matches are approximately the same distance apart and have the same relative orientation differences. The sets of consistent hypotheses then are used to find a the pose for verification. Finally the hypothesized model and pose is verified with the scene data. They have shown that in the best case, where only one object is present in the scene, the complexity can be as low as $O(n)$, but in the worst case where multiple instances of the the object are present in the scene with partial occlusion then the complexity can be as high as $O(n^2m^3)$, where n is the number of scene feature points and m is the number of models. They have shown through experimentation that the system works fairly quickly and can handle complex scenes with moderate amounts of noise.

Chua and Jarvis [25] formulated a new representation (the point signature), which follows along the same lines as Stein and Medioni's work [105]. The point signature is different in that it does not encode information about the normals around the points of interest (Fig. 13); rather it encodes the minimum distances of points on a 3D contour to a reference plane (Fig. 15). The contour is constructed by intersecting the surface with a sphere centered on P and with a fixed radius r. A principal component analysis of the 3D contour defines a plane where the distance from points on the contour to the plane are at a minimum. The normal of the plane can be thought of as approximating the surface normal around the point of interest. The plane is then translated until it contains P. A signed distance from the 3D contour forms a 1D parametric curve $d(\theta)$ as shown in Fig. 15. The final representation is a discretized version $d[n] = d(n * \Delta\theta)$, where $\Delta\theta$ is 15°. This provides a compact way of storing information about the structure of the surface around a point that is pose invariant. For their final system they found that it was better to encode two signatures at every point of interest on the object where the two signature were created with spheres of different radii. This improved the selectivity of the matches.

To sufficiently cover the object, a discrete parametric curve $d[n]$ is obtained at every model point. These point signatures are then placed in an index table. Each bin in the table contains a list of model point signatures whose min $d[n]$ and max $d[n]$ values are

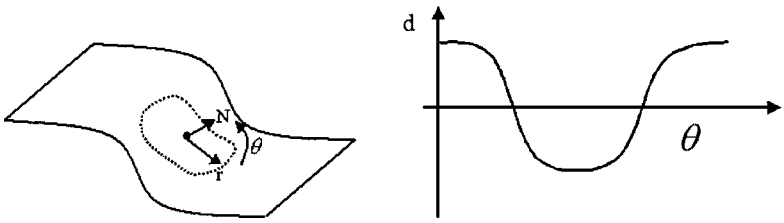

FIG. 15. Point signature.

similar. During recognition, point signatures are calculated on a grid evenly spaced over the scene. These point signatures are used to index the table and compare with the model signatures contained in the appropriate bin. Signatures that are similar generate possible model scene correspondences. The hypotheses are grouped by model and models are ordered by the number of hypotheses they received. The models with the most correspondences are verified. The rotation and translation transformation between the scene point and model are found using partial correspondence search [24, 25]. The worst case computational cost of the verification is $O(N_s N_m H^3)$, where N_s is the number of scene points in the hypotheses, N_m is the number of model points in the correspondence, and H is the maximum number of hypotheses a scene point is mapped to model points (since a scene point can map to more than one model point). The system shows quick recognition times for multiple-object scenes from a database of 15 models with high accuracy.

Johnson and Hebert [57, 54, 56, 53, 59, 60] also employed point features for object recognition. Their "spin images" are 2D histograms of the surface locations around a point. The spin image is generated using the normal to the point and rotating a cutting plane around the point using the normal as the axis of rotation (see Fig. 16). As the plane spins around the point the intersections between the plane and the surface are used to index a 2D histogram. The bins in the histogram represent the amount of surface (or the number of times) a particular patch of the cutting plane intersects the object. Spin images are generated from models using a polygonal approximation to the surface. The vertices in the polygonal model are approximately uniformly distributed across the object. When spin images are generated from real data taken from a range scanner, a similar criterion is required. The uniformity is required so the bins in the spin images approximate the surface.

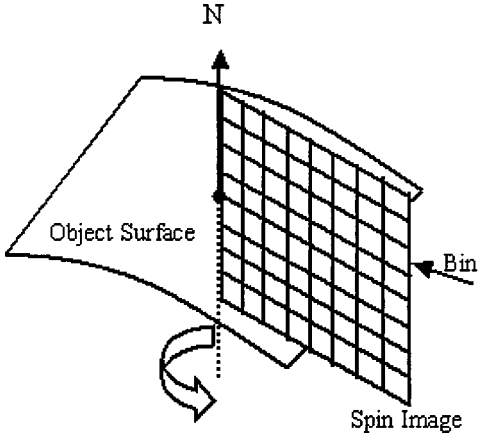

FIG. 16. Spin image.

Johnson and Hebert's [56, 53] spin images are used to establish correspondence between scene and model points for 3D data. During recognition the scene is randomly sampled to find points where spin images are to be found. This is done until 10% of the scene points have been sampled. For each scene spin image is compared with every model spin image. The comparisons produce a similarity measure histogram. The histogram is analyzed to find the upper outliers. These outliers are used to generate hypothetical matches between a scene point and possibly more than one model point. Once all the scene points have been used to generate possible correspondences, they are again filtered to remove 50% of the worst matches. The correspondences are then grouped by model and checked for geometric consistency. The measure ensures consistency in both geometric position and orientation. The measure is weighted by distance between the sets of correspondences. This is used to promote grouping of matches that are geometrically consistent and are separated by some distance. Groups of geometrically consistent matches between model and scene points are used to calculate a rotation and translation transformation between model and scene points. A modified ICP algorithm [10] is used to refine the transformation before a final verification step confirms or rejects the model/scene match. The recognition scheme can be used to identify partially occluded free-form objects. The system is shown to work on an object where only 20% of the surface is visible. The space requirements of the spin image stack is $O(MI)$ where M is the number of model points (this includes all models in the database) and I is the number of pixels (bins) in the spin images. The computational complexity for establishing correspondences is $O(SMI + SM\log(M))$, where S is the number of randomly sampled scene points, and the term $M\log(M)$ identifies time needed to sort the similarity measures between a scene point and all the model points in the spin image stack. The complexity of grouping the correspondences is not given, but the ICP refinement of the transformation is at worst $O(M)$ for each iteration.

Johnson and Hebert [59] have also adapted the appearance-based techniques of Murase and Nayar [75] for use with the spin images. The images are used to find a low-dimensional subspace where scene spin images can be match quickly to model spin images. New appearance-based representation for the discriminatory database of spin images significantly reduced the amount of storage space, while enabling the system to more effectively recognize multiple objects in one scene range image, but their results have shown a slight decrease in the recognition rate when compared with previous recognition schemes [56, 53]. Their testing further shows that the amount of clutter in the scene does not seem to affect the recognition rate, while the amount of occlusion of the object does. It seems that if a sufficient amount (more than 30%) of the object surface is visible in the scene, it has a high probability of being recognized.

Spherical representations for representing 3D surfaces for object recognition and pose localization have a rich history in the vision community [68, 27, 116, 98, 45, 52, 29]. Ikeuchi and Hebert [52] provide an excellent overview of the use of these representations from the early 1980s to the present. Delingette *et al*.'s recently developed spherical attribute image (SAI) [29, 45] to address many of the shortcomings of the earlier spherical representations (EGI, DEGI, and CEGI [52]). Those problems are ability to handle nonconvex objects (representation has a many-to-one mapping), and the ability to handle occlusion (partial surface matching).

The SAI representation maps points on an object surface to vertices on a quasi-regular tessellated sphere. By *regular* it is implied that the triangle faces are similar in surface area and equilateral. Local surface characteristics are stored at the vertices of the tessellated

sphere that correspond with the surface point. The surface point to vertex mapping is determined by deforming/shrinking an ellipsoidal version of the same tessellated sphere to the object's surface. The deformation is guided by forces that try to preserve the regularity, while shrinking the mesh to the surface. This regularity condition gives the SAI representation rotational invariance and the ability to be extended to match-occluded objects.

Delingette *et al*. [29, 45] point out that the rotational invariance is only true as the number of nodes in the mesh becomes very large, because the SAI is discrete and the nodes of the mesh are mapped to the object's surface. They overcome this problem by averaging SAI values at vertices to approximate values at any point in between the mesh nodes.

Occlusion is handled by assuming the mesh faces have equal area and using a scale factor to enlarge or shrink the SAI deformed to a partial surface. In the generation of the SAI, a closed surface mesh is deformed to the partial surface; this produces some areas on the SAI where the mesh is interpolating the space in between surface data from the partial data of the object. These interpolated regions will not be matched with another SAI. Typically, the SAI generated from only part of an object surface will not map to the same number of mesh nodes as one fit to the full object. To allow such spherical attribute images to be matched, one mesh must be shrunk or enlarged so that, when matched, they represent the same amount of area on the objects they model.

Matching two objects reduces to finding the best rotation between scene and model SAIs. The simplest strategy is to sample the space of all possible rotations of the sphere and to evaluate the distance measure between SAIs. The rotation that minimizes the error is a candidate match. For more than one object in the database, each object SAI must be matched with the scene. To reduce the search space for the best rotation the geometry of the tessellated sphere is used to limit the number of possible rotations. The best match can be determined in a few seconds on a modern workstation. This approach has been extended to include occluded objects. The mesh resulting from a partial view of an object is scaled and the interpolated nodes do not count in the distance measures between SAIs. A shape similarity measure has been introduced between SAIs so classification of similar objects can be used or a hierarchical database. Sets of similar objects can be classified together under a prototype for the class. Similarity is done at different scales of smoothing of the SAI. A multiscale version of an object's SAI then can be matched with the prototype to determine if the objects in the class should be matched against the scene SAI. This can reduce the number of times the scene SAI must be matched against database objects.

In the work summarized above, objects must have the same topology as a sphere. Delingette [27] uses a representation similar to the SAI to represent more complex topologies. Instead of maintaining the relationship with a sphere, parts of an object can be fitted using the deformable surface mesh and can be connected to another mesh. The use of this representation for object recognition and localization of pose has not yet been explored.

Shum *et al*. [98] developed a similarity measure for spherical attribute images. The similarity measure is used to show the SAI's ability to differentiate objects and how the number of nodes in the tessellation affects matching of objects. Zhang and Hebert [116] also use the similarity measure to classify an object's SAI at different scales. They have also added the notion of smoothing an object's surface through its SAI representation without shrinkage. This has allowed the matching of objects at different levels of detail.

Barequet and Sharir [5] formulated a novel approach for partial surface and volume matching by finding registration parameters. The registration technique was inspired by a intensity image technique for matching and pose localization. The method uses a footprint

that is invariant to rotation and translation to differentiate objects. The footprints change depending on the application. One example of a footprint is an approximation of surface curvature at a point, which is invariant to pose. Using the footprints, correspondences are established between scene and model points. The list of correspondences is passed through a series of discrete rotations and used to score the quality of the match. This produces a voting table where the entries record the quality of a rotation. The entry in the table with the best score is used as an initial rotation estimate for the object and fed to an iterative refinement algorithm. Once the best rotation has been determined, the translation transformation is found. The complexity of finding the match is $O(n + k + s^3)$, where n is number of points in the input sets, k is the number of correspondences, and s is the maximum size of the voting table.

Greenspan [43] used a hypothesis-test-verify methodology to avoid the typical feature extraction techniques associated with 3D recognition. The method's objective is to determine all occurrences of a model in the image. This is accomplished by using image seed points as starting points for a sequential hypothesis-and-test algorithm. The seed point is hypothesized to lie on the surface of the model. The set of possible poses that maintain the truth of this hypothesis is large since the point can be any point on the object. To reduce the number of possible poses, successive image points are queried and their hypotheses intersected with the current set until only a small number of hypotheses are left. At each stage the new point is hypothesized to lie on the surface on the model and tested. This is accomplished efficiently by generating a sample tree from the model and using it to guide the choice of points around the seed point. Since most seed point combinations will not match well to a model, the tree is designed to be efficient at refuting incorrectly the hypotheses. In the implementation of the system, the image is used to generate a coarse voxel representation of the data. This is done for two reasons: first it is easy to determine if a point in space is close to the surface of the object (voxels are labeled as surface, occluded, free, unknown) and it effectively limits the number of possible model poses. The time complexity to traverse the sample tree with n leaf nodes for s image seeds is $O(s \log^2 n)$. This is the time it takes to generate a set of initial pose hypotheses for an object in a scene image. The hypotheses are checked using template correlation between model and scene data for a given pose. The surviving hypotheses are further eliminated by looking at the amount of overlapping surface area and the volume between the surfaces in that area. Finally the few hypotheses left are sent to an ICP algorithm [10] to refine the pose and tested again for the overlap area and the volume between the surfaces. The sample trees took several days to generate on a modern workstation, while recognition was achieved in about a minute. The system was shown to handle recognition and pose determination of free-form objects in complex scenes with occlusion.

Complex objects with similar gross shape but different fine shape detail present new and interesting problems to object recognition practice. One example of this is the human brain, which has similar overall shape but unique convolution detail. For a certain class of problems, a general shape descriptor would allow studies of common properties of the object without being adversely affected by individual variations of single entities. This is in contrast to another class of problems where one might want to identify differences between an individual and other members in its population. In the first class of problems the general shape and large structures are important, while in the second class of problems the individual variations are important. To handle both of these cases, an object must represent both local and global details. Naf *et al.* [76] proposed using 3D Voronoi skeletons to represent medical volumetric data. A Voronoi skeleton is derived by first finding the 3D Voronoi diagram of the

TABLE 3
Summary of Recognition Techniques for Range Data

Technique	Features	Occlusion local	Largest database	Recognition rate	Complexity
Nevatia and Binford [81]	Generalized cones	Yes	5	NR	NR
Raja and Jain [91]	Geons	No	36 synthetic and 12 real	0.77	NR
Extremal mesh [110]	Curvature	No	N/A	NR	NR
COSMOS [31]	Curvature (shape spectra)	No	20 synthetic and 10 real	0.97	NR
Tripod [87]	Crease angle	Yes	4	0.92	NR
Splash [105]	Normal (structural indexing)	Yes	9	NR	$O(n) - O(n^2m^3)$
Point signature [25]	Distance	Yes	15	1.0	Worst case $O(N_s N_m H^3)$
Spin image [60]	Surface histogram	Yes	4	1.0	$O(k \log_2(n))$
SAI [45]	Angle	Yes	N/A	NR	NR
Barequet and Sharir [5]	Curvature	Yes	N/A	NR	$O(n + k + s^3)$
Greenspan [43]	Sample tree	Yes	5	NR	$O(s \log^2(n))$
Naf *et al.* [76]	Voronoi skeleton	No	N/A	NR	NR

volumetric data then iteratively removing the Voronoi faces that are closest to the object's boundary and whose removal will not change the topology of the diagram. When no more faces can be removed, what remains is a 3D Voronoi skeleton of the object. At this level, the diagram represents the general shape of the object, compared to the original Voronoi diagram of the data, which contains the individual characteristics. The Voronoi skeletons where used to find the thickest part of a hip bone from a 3D radiological scan of the hip and to analyze abnormalities in the temporal lobe of a brain MRI.

Table 3 summarizes and compares the various techniques described above. The table includes information about the geometric feature the technique is built upon (e.g., surface normal or curvature, geons, generalized cones), and whether the technique can handle object occlusion. The table also includes the largest documented database size and the recognition rate and time complexity of the algorithm. Not all the papers surveyed reported recognition rate or the complexity of the recognition algorithm. In these cases the table is marked with an NR (not reported). In some papers the method was mostly described as a surface- or volume-matching technique. In these cases often what was described was the ability of the system to accurately match two data sets by reporting the recovered pose vs known pose. For these papers the largest database size was listed as N/A to indicate the tests were not designed to discriminate between objects, but rather to recover correct pose of a known object in the scene.

6. EMERGING THEMES, CONCLUSIONS, AND COMMENTARY

The focus of this survey has been the construction and recognition of three-dimensional object models, with primary emphasis on range data but with some mention of intensity

techniques as context. Toward that end, modeling and recognition are both surveyed in detail.

Improvements in modeling technique will increase the accuracy and speed of reverse engineering complex 3D models from examples. The quality of these models at present depends on the skill of the person monitoring and manipulating the software packages used to form the model. The designer must currently be constantly on the watch for erroneous data (errors in measurement by the sensor), errors in the registration process, and the integrity of the mesh after integration. Each one of these tasks may and often does fail when there are errors made in the measurement of the object and/or the position and amount of overlap between surfaces. The methods developed for model building have only been tested on a small set of objects. This is in part due to the large effort required to train an algorithm to work on a set of objects. Often the heuristics used to produce good results on one model will fail on another. In the future the methods for registration, integration, and optimization of polygonal meshes need to be tested on a large number of standardized objects with varying geometry so the strengths and weaknesses can be cataloged for each technique. The research community would benefit from an "open source" architecture and implementation of the standard techniques used in range image registration and integration. This would be a valuable resource that allows the community to build on the knowledge of the previous researchers without wasting time and effort in rediscovering what has already been done.

The object recognition problem involves a study of salient features and their identification, as well as a study of control and data structures that yield efficient recognition techniques. The problem of finding and identifying objects in single-object scenes with no occlusion has been well studied and many systems designed show good results [3, 13, 31, 52, 75, 77, 113]. In these systems a fair amount of information is present about the object. This allows for the design of systems that are able to identify objects under noisy conditions except in cases where views of the objects are too similar. In these cases noise can dominate the differences between the very similar object views and cause the reliability of the recognizer to decrease. In the future these ambiguous views need to be determined and documented for each model and system so that a designer can determine whether these difficult views are significant for an application. This is especially important as more and more objects are added to a system and the chances of similarity increase.

The problem of finding and identifying multiple objects in scenes with the possibility of occlusion and background clutter is a much harder problem [12, 21, 25, 43, 45, 59, 66, 84, 105]. In general, the partial information about an object makes the recognition less reliable and more complex because the information can be incomplete and disjoint. This is in part due to the decreased saliency of local measures over global measures. Locally many regions of a surface or many regions of surfaces on different objects may appear similar. Using the information in these regions by themselves is a poor choice for recovering the identity and location, but together by using the relationships between features the identity and location of an object may still be obtained. In the future the measures of saliency for features in the illuminance, color, and range image domains need to be qualified in terms of their sensitivity to noise so that their discriminatory properties can be more clearly specified and compared.

For both single- and multiple-object scenes large database studies need to be applied. As the size of the database increases the importance of a system's method for quickly and accurately indexing to the correct model becomes more important. Large multimedia

databases and flexible manufacturing inspection and assembly systems are examples of applications where quick indexing of image/model databases are becoming important.

An emerging area of study for new object recognition systems is to combine multiple imaging modalities to determine object identity and location. One example is to use both depth (range) and color information from many modern 3D digitizers common to the computer graphics and computer vision fields. The textural information from the color images may allow for discrimination between objects in cases where their shape is similar while their texture is different; likewise cases where the textural information does not discriminate between objects well, their shapes may.

Another area likely to increase in importance is deformable object modeling and understanding. Techniques to deal with nonrigid object matching and motion are an important field especially in the areas of medical imaging. Here the individuality of a patient may cause ridge techniques problems because the surface geometry of an organ or other parts varies from person to person. Even in the case of maintaining a health history of a single patient the parts can change due to swelling or aging. Therefore what may be more interesting for nonrigid objects is to study their defining characteristics so that time-dependent shapes may be represented and recovered faithfully.

ACKNOWLEDGMENTS

This work has been supported by the National Science Foundation under Grants IRI-9209212, IRI-9506414, IIS-9996004, CDA-9422044, and EIA-9818212, by the Department of Electrical Engineering at the Ohio State University and by the School of Electrical Engineering and Computer Science at Washington State University. The authors thank three anonymous reviewers for helpful comments and suggestions.

REFERENCES

1. J. Agin and T. O. Binford, Computer description of curved objects, *IEEE Trans. Comput*. **25**, 1976, 439–449.
2. A. P. Ashbrook, R. B. Fisher, C. Robertson, and N. Werghi, Construction of articulated models from range data, in *Proc. 1999 British Machine Vision Conference (BMVC '99), Nottingham, 1999*, pp. 183–192.
3. N. Ayong-Chee, G. Dudek, and F. P. Ferrie, A hybrid approach to 3D representation, in *Object Representation in Computer Vision II* (M. Hebert, J. Ponce, and A. Zisserman, Eds.), pp. 321–333, Springer-Verlag, Berlin, 1996.
4. C. L. Bajaj, J. Chen, and G. Xu, Modeling with cubic A-patches, *ACM Trans. Graphics* **14**, 1995, 103–133.
5. G. Barequet and M. Sharir, Partial surface and volume matching in three dimensions, *IEEE Trans. Pattern Anal. Mach. Intell*. **19**, 1997, 929–948.
6. A. H. Barr, Superquadrics and angle preserving transformations, *IEEE Compt. Graphics Appl*. **1**, 1981, 11–23.
7. R. Bergevin, M. Soucy, H. Gagnon, and D. Laurendeau, Towards a general multi-view registration technique, *IEEE Trans. Pattern Anal. Mach. Intell*. **18**, 1996, 540–547.
8. P. J. Besl, Triangles as a primary representation, in *Object Representation in Computer Vision* (M. Hebert, J. Ponce, T. Boult, and A. Gross, Eds.), pp. 191–206. Springer-Verlag, Berlin, 1995.
9. P. J. Besl and R. C. Jain, Three-dimensional object recognition, *Comput. Surv*. **17**, 1985, 75–145.
10. P. J. Besl and N. D. McKay, A method for registration of 3-D shapes, *IEEE Trans. Pattern Anal. Mach. Intell*. **14**, 1992, 239–256.
11. P. J. Besl, The free-form surface matching problem, in *Machine Vision for Three-Dimensional Scences* (H. Freeman, Ed.), pp. 25–71. Academic Press, San Diego, 1990.

12. H. Bischof and A. Leonardis, Robust recognition of scaled eigenimages through a hierarchical approach, in *Proc. IEEE Conf. Computer Vision and Pattern Recognition, Santa Barabra, California, June 1998*, pp. 664–670.
13. M. J. Black and A. D. Jepson, Eigen tracking: Robust matching and tracking of articulated objects using a view-based representation, *Int. J. Comput. Vision* **26**, 1998, 63–84.
14. G. Blais and M. D. Levine, Registering multiview range data to create 3D computer objects, *IEEE Trans. Pattern Anal. Mach. Intell*. **17**, 1995, 820–824.
15. J. P. Brady, N. Nandhakumar, and J. K. Aggarwal, Recent progress in the recognition of objects from range data, *Image Vision Comput.* **7**, 1989, 295–307.
16. C. M. Brown, Some mathematical and representational aspects of solid modeling, *IEEE Trans. Pattern Anal. Mach. Intell*. **3**, 1981, 444–453.
17. L. G. Brown, A survey of image registration techniques, *ACM Comput. Surv*. **24**, 1992, 325–376, December 1992.
18. J. Weng, C. Dorai, and A. K. Jain, Optimal registration of object views using range data, *IEEE Trans. Pattern Anal. Mach. Intell*. **19**, 1997, 1131–1137.
19. R. Campbell and P. Flynn, Model and range image features for free-form object recognition, in *Vision Interface, Vancouver, BC, June 1998*, pp. 173–180.
20. R. Campbell and P. Flynn, Eigenshapes for 3D object recognition in range data, in *Proc. IEEE Conf. Computer Vision and Pattern Recognition, Fort Collins, Colorado, June 1999*.
21. O. I. Camps, C. Y. Huang, and T. Kanungo, Hierarchical organization of appearance-based parts and relations for object recognition, in *Proc. IEEE Conf. Computer Vision and Pattern Recognition, Santa Barabra, California, June 1998*, pp. 685–691.
22. J. L. Chen and G. C. Stockman, 3D free-form object recognition using indexing by contour features, *Comput. Vision Image Understand*. **71**, 1998, 334–335.
23. Y. Chen and G. Medioni, Object modeling by registration of multiple range images, *Image Vision Comput.* **10**, 1992, 145–155.
24. C. S. Chua and R. Jarvis, 3D free-form surface registration and object recognition, *Int. J. Comput. Vision* **17**, 1996, 77–99.
25. C. S. Chua and R. Jarvis, Point signatures: A new representation for 3D object recognition, *Int. J. Comput. Vision* **25**, 1997, 63–85.
26. J. Cohen, D. Manocha, and M. Olano, Simplifying polygonal models using successive mappings, in *Proceedings IEEE Visualization '97, 1997*, pp. 395–402.
27. H. Delingette, Simplex meshes: A general representation for 3D shape reconstruction, in *Proc. IEEE Conf. Computer Vision and Pattern Recognition, Seattle, Washington, June 1994*, pp. 856–859.
28. H. Delingette, M. Hebert, and K. Ikeuchi, Shape representation and image segmentation using deformable surfaces, *Image Vision Comput*. **10**, 1992, 132–144.
29. H. Delingette, M. Hebert, and K. Ikeuchi, A spherical representation for the recognition of curved objects, in *Proc. IEEE Int. Conf. On Computer Vision, Berlin, Germany, May 1993*, pp. 103–112.
30. P. Dierckx, *Curve and Surface Fitting with Splines*, Oxford Science, New York, 1993.
31. C. Dorai and A. K. Jain, COSMOS—A representation scheme for 3D free-form objects, *IEEE Trans. Pattern Anal. Mach. Intell*. **19**, 1997, 1115–1130.
32. J. Edwards and H. Murase, Appearance matching of occluded objects using coarse-to-fine adaptive masks, in *Proc. IEEE Conf. Computer Vision and Pattern Recognition, San Juan, Puerto Rico, June 1997*, pp. 533–539.
33. D. W. Eggert, A. W. Fitzgibbon, and R. B. Fisher, Simultaneous registration of multiple range views for use in reverse engineering of CAD models, *Comput. Vision Image Understand*. **69**, 1998, 253–272.
34. S. J. Dickinson *et al*., Panel report: The potential of geons for generic 3-D object recgnition, *Image Vision Comput*. **15**, 1997, 277–292.
35. G. Farin, *Curves and Surfaces For CAGD*, third ed., Academic Press, San Diego, 1993.
36. P. Flynn and A. K. Jain, Bonsai: 3D object recognition using constrained search, *IEEE Trans. Pattern Anal. Mach. Intell*. **13**, 1992, 1066–1075.

37. P. J. Flynn and A. K. Jain, Three-dimensional object recognition, in *Handbook of Pattern Recognition and Image Processing: Computer Vision*, Academic Press, San Diego, 1994, pp. 497–541.
38. J. D. Foley, A. Van Dam, S. K. Fiener, J. F. Hughes, and R. L. Phillips, *Introduction to Computer Graphics*, Addision-Wesley, Reading, MA, 1994.
39. J. H. Friedman, J. L. Bentley, and R. A. Finkel, An algorithm for finding best matches in logarithmic expected time, *ACM Tran. Mathematical Software* **3**, 1977, 209–226.
40. H. Gagnon, M. Soucy, R. Bergevin, and D. Laurendeau, Registration of multiple range views for automatic 3-D model building, in *Proc. IEEE Conf. Computer Vision and Pattern Recognition, Seattle Washington*, June 1994, pp. 581–586.
41. M. Garland and P. S. Heckbert, Surface simplification using quadric error metrics, in *Proceedings of SIGGRAPH '97*, 1997.
42. K. Green, D. Eggert, L. Stark, and K. Bowyer, Generic recognition of articulated objects through reasoning about potential function, *Comput. Vision Image Understand*. **62**, 1995, 177–193.
43. M. Greenspan, The sample tree: A sequential hypothesis testing approach to 3D object recognition, in *Proc. IEEE Conf. Computer Vision and Pattern Recognition, Santa Barabra, California, June 1998*, pp. 772–779.
44. A. Gupta and R. Bajcsy, Volumetric segmentation of range images of 3D objects using superquadric Models, *CVGIP: Image Understand*. **58**, 1991, 302–326.
45. M. Hebert, K. Ikeuchi, and H. Delingette, A spherical representation for recognition of free-form surfaces, *IEEE Trans. Pattern Anal. Mach. Intell*. **17**, 1995, 681–690.
46. P. Heckbert and M. Garland, Survey of polygonal surface simplification algorithms, in *Multiresolution Surface Modeling Course SIGGRAPH '97*, 1997.
47. H. Hoppe, Progressive meshes, in *Proceedings of SIGGRAPH '96*, pp. 99–108, 1996.
48. H. Hoppe, New quadric metric for simplifying meshes with appearance attributes, in *Proceedings IEEE Visualization '99*, pp. 24–29, 1999.
49. H. Hoppe, T. DeRose, and T. Duchamp, Mesh optimization, in *Proceedings of SIGGRAPH '93*, pp. 19–26, 1993.
50. H. Hoppe, T. DeRose, T. Duchamp, J. McDonald, and W. Stuetzle, Surface reconstruction from unorganized points, in *Proceedings of SIGGRAPH '92*, pp. 71–78, 1992.
51. C. Y. Huang, O. I. Camps, and T. Kanungo, Object recognition using appearance-based parts and relations, in *Proc. IEEE Conf. Computer Vision and Pattern Recognition, San Juan, Puerto Rico, June 1997*, pp. 877–883.
52. K. Ikeuchi and M. Hebert, Spherical representations: From EGI to SAI, in *Object Representation in Computer Vision* (M. Hebert, J. Ponce, T. Boult, and A. Gross, Eds.), pp. 327–345, Springer-Verlag, Berlin, 1995.
53. A. E. Johnson and M. Hebert, Recognizing objects by matching oriented points, Robitics Institute CMU-RI-TR-96-04, Carnegie Mellon University, May 1996.
54. A. E. Johnson and M. Hebert, A system for semi-automatic modeling of complex environments, in *Proc. IEEE Int. Conf. On Recent Advances in 3-D Digital Imaging and Modeling Ottawa, Canada, May 1997*, pp. 213–220.
55. A. E. Johnson and M. Hebert, Control of polygonal mesh resolution for 3-D computer vision, Technical Report CMU-RI-TR-96-20, Carnegie Mellon University, April 1997.
56. A. E. Johnson and M. Hebert, Recognizing objects by matching oriented points, in *IEEE Conf. Computer Vision and Pattern Recognition, San Juan, Puerto Rico, June 1997*, pp. 684–689.
57. A. E. Johnson and M. Hebert, Surface registration by matching oriented points, in *Proc. IEEE Int. Conf. On Recent Advances in 3-D Digital Imaging and Modeling, Ottawa, Canada, May 1997*, pp. 121–128.
58. A. E. Johnson and M. Hebert, Control of polygonal mesh resolution for 3-D computer vision, *Graphical Models Image Process*. **60**, 1998, 261–285.
59. A. E. Johnson and M. Hebert, Efficient multiple model recognition in cluttered 3-D scenes, in *Proc. IEEE Conf. Computer Vision and Pattern Recognition, Santa Barabra, California, June 1998*, pp. 671–677.
60. A. E. Johnson and M. Hebert, Surface matching for object recognition in complex three-dimensional scenes, *Image Vision Comput*. **16**, 1998, 635–651.

61. T. Joshi, J. Ponce, B. Vijayakumar, and D. J. Kriegman, HOT curves for modeling and recognition of smooth curved 3D objects, in *IEEE Conf. Computer Vision and Pattern Recognition, Seattle, Washington, June 1994*, pp. 876–880.
62. T. Joshi, B. Vijayakumar, D. Kriegman, and J. Ponce, HOT curves for modelling and recognition of smooth curved 3D objects, *Image Vision Comput*. **15**, 1997, 479–498.
63. M. Kirby and L. Sirovich, Application of the Karhunen-Loeve procedure for the characterization of human faces, *IEEE Trans. Pattern Anal. Mach. Intell*. **12**, 1990, 103–108.
64. V. Koivunen and R. Bajcsy, Spline representations in 3-D vision, in *Object Representation in Computer Vision*, (M. Hebert, J. Ponce, T. Boult, and A. Gross Eds.), pp. 177–190, Springer-Verlag, Berlin, 1995.
65. V. Krishnamurthy and M. Levoy, Fitting smooth surfaces to dense polygon meshes, in *Proceedings of SIGGRAPH '96 New Orleans, Louisiana, August 1996*, pp. 313–324.
66. J. Krumm, Eigenfeatures for planar pose measurement of partially occluded objects, in *Proc. IEEE Conf. Computer Vision and Pattern Recognition, San Francisco, California, June 1996*, pp. 55–60.
67. A. Leonardis and H. Bischof, Dealing with occlusions in the eigenspace approach, in *Proc. IEEE Conf. Computer Vision and Pattern Recognition, San Francisco, California, June 1996*, pp. 453–458.
68. Y. Li and R. J. Woodham, Orientation-based representations of 3-D shape, in *Proc. IEEE Conf. Computer Vision and Pattern Recognition, Seattle, Washington, June 1994*, pp. 182–187.
69. P. Lindstrom and G. Turk, Fast and memory efficient polygonal simplification, in *Proceedings IEEE Visualization '98*, pp. 279–286, 1998.
70. W. E. Lorenson and H. E. Cline, Marching cubes: A high resolution 3D surface construction algorithm, in *Computer Graphics, Proceedings of SIGGRAPH '87*, Vol. 21, pp. 163–169, 1987.
71. F. Mokhtarian, Silhouette-based isolated object recognition through curvature scale space, *IEEE Trans. Pattern Anal. Mach. Intell*. **17**, 1995, 539–544.
72. S. Mukherjee and S. K. Nayar, Automatic generation of RBF networks, Technical Report CUCS-001-95, Columbia University, 1995.
73. J. Mundy, A. Liu, N. Pillow, A. Zisserman, S. Abdallah, S. Utcke, S. Nayar, and C. Rothwell, An experimental comparison of appearance and geometric model based recognition, in *Object Representation in Computer Vision II* (J. Ponce, A. Zisserman, and M. Hebert, Eds.), pp. 247–269. Springer-Verlag, Cambridge, UK, 1996.
74. H. Murase and S. K. Nayar, Illumination planning for object recognition using parametric eigenspaces, *IEEE Trans. Pattern Anal. Mach. Intell*. **16**, 1994, 1219–1227.
75. H. Murase and S. K. Nayar, Visual learning and recognition of 3-D objects from appearance, *Int. J. Comput. Vision* **14**, 1995, 5–24.
76. M. Naf, G. Szekely, R. Kikinis, M. E. Shenton, and O. Kubler, 3D Voronoi skeletons and their usage for the characterization and recognition of 3D organ shape, *Comput. Vision Image Understand*. **66**, 1997, 147–161.
77. S. K. Nayar, S. A. Nene, and H. Murase, Real-time 100 object recognition system, in *Proc. Of ARPA Image Understanding Workshop, ARPA, Palm Springs, February 1996*.
78. S. A. Nene and S. K. Nayar, A simple algorithm for nearest neighbor search in high dimensions, Technical Report CUCS-030-95, Columbia University, Department of Computer Science, 1995.
79. S. A. Nene and S. K. Nayar, A simple algorithm for nearest-neighbor search in high dimensions, *IEEE Trans. Pattern Anal. Mach. Intell*. **19**, 1997, 989–1003.
80. P. J. Neugebauer, Geometrical cloning of 3D objects via simultaneous registration of multiple range images, in *Proc. IEEE Inter. Conf. Shape Modeling and Applications, Aizu-Wakamatsu, Japan, March 1997*, pp. 130–139.
81. R. Nevatia and T. O. Binford, Descrition and recognition of curved objects, *Artif. Intell*. **8**, 1977, 69–76.
82. P. Ning and J. Bloomenthal, An evaluation of implicit surface tilers, *Comput. Graphics Appl*. **13**, 1993, 33–41.
83. C. Oblonsek and N. Guid, A fast surface-based procedure from object reconstruction from 3D scattered points, *Comput. Vision Image Understand*. **69**, 1998, 185–195.
84. K. Ohba and K. Ikeuchi, Detectability, uniqueness, and reliability of eigen windows for stable verification of partially occluded objects, *IEEE Trans. Pattern Anal. Mach. Intell*. **19**, 1997, 1043–1048.

85. J. Park, D. Metaxas, and A. Young, Deformable models with parameter function: Application to heart-wall modeling, in *Proc. IEEE Conf. Computer Vision and Pattern Recognition, Seattle, Washington, June 1994*, pp. 437–422.

86. A. P. Pentland, Perceptual organization and the representation of natural form, *Artif. Intell*. **28**, 1986, 293–331.

87. F. Pipitone and W. Adams, Rapid recognition of freeform objects in noisy range images using tripod operators, in *Proc. IEEE Inter. Conf. On Computer Vision, Berlin, Germany, May 1993*, pp. 715–716.

88. J. Ponce and D. J. Kriegman, On recognizing and positioning curved 3D objects from image contours, in *IEEE Workshop on Interpretation of 3D Scenes, Austin, Texas, November 1989*, pp. 61–67.

89. A. R. Pope, Model-based object recognition: A survey of recent research, Technical Report 94-04, Univ. of British Columbia, January 94.

90. K. Pulli, Multiview registration for large data sets, in *Proc. 1999 Conf on 3D Digital Imaging and Modeling (3DIM '99)*, pp. 160–168, 1999.

91. N. S. Raja and A. K. Jain, Recognizing geons from superquadrics fitted to range data, *Image Vision Comput*. **10**, 1992, 179–190.

92. R. Rao, Dynamic appearance-based recognition, in *Proc. IEEE Conf. Computer Vision and Pattern Recognition, San Juan, Puerto Rico, June 1997*, pp. 540–546.

93. M. K. Reed and P. K. Allen, 3-D modeling from range imagery: An incremental method with a planning component, *Image Vision Comput*. **17**, 1999, 99–111.

94. M. K. Reed, P. K. Allen, and I. Stamos, Automated model acquisition from range images with view planning, in *Proc. IEEE Conf. Computer Vision and Pattern Recognition, San Juan, Puerto Rico, June 1997*, pp. 72–77.

95. M. Sallam and K. Bowyer, Generalizing the aspect graph concept to include articulated assemblies, *Pattern Recognit. Lett*. **12**, 1991, 171–176.

96. H. Samet, *Applications of Spatial Data Structures*, Addison-Wesley, Reading, MA, 1990.

97. J. Shen and Y.-H. Yang, Deformable object modeling using the time-dependent finite element method, *Graphical Models Image Process*. **60**, 1998, 461–487.

98. H.-Y. Shum, M. Hebert, and K. Ikeuchi, On 3D shape similarity, in *Proc. IEEE Conf. Computer Vision and Pattern Recognition, San Francisco, California, June 1996*, pp. 526–531.

99. H-Y Shum, M. Hebert, K. Ikeuchi, and R. Reddy, An integral approach to free-form object modeling, *IEEE Trans. Pattern Anal. Mach. Intell*. **19**, 1997, 1366–1370.

100. F. Solina and R. Bajcsy, Recovery of parametric models from range images: The case for superquadrics with global deformations, *IEEE Trans. Pattern Anal. Mach. Intell*. **12**, 1990, 131–147.

101. M. Soucy, G. Godin, and M. Rioux, A texture-mapping approach for the compression of colored 3d triangulations, *Visual Comput.* **12**, 1996, 503–514.

102. M. Soucy and D. Laurendeau, A dynamic integration algorithm to model surface from multiple range views, *Mach. Vision Appl*. **8**, 1995, 53–62.

103. M. Soucy and D. Laurendeau, A general surface approach to the integration of a set of range views, *IEEE Trans. Pattern Anal. Mach. Intell*. **17**, 1995, 344–358.

104. M. Soucy and D. Laurendeau, Multiresolution surface modeling based on hierarchical triangulation, *Comput. Vision Image Understand*. **63**, 1996, 1–14.

105. F. Stein and G. Medioni, Structural indexing: Efficient 3-D object recognition, *IEEE Trans. Pattern Anal. Mach. Intell*. **14**, 1992, 125–145.

106. J. Subrahmonia, D. B. Cooper, and D. Keren, Practical reliable Bayesian recognition of 2D and 3D objects using implicit polynomials and algebraic invariants, *IEEE Trans. Pattern Anal. Mach. Intell*. **18**, 1996, 505–519.

107. S. Sullivan, L. Sandford, and J. Ponce, Using geometric distance fits for 3-D object modeling and recognition, *IEEE Trans. Pattern Anal. Mach. Intell*. **16**, 1994, 1183–1195.

108. G. Taubin, Estimation of planar curves, surfaces, and nonplanar space curves, defined by implicit equations with applications to edge and range image segmentation, *IEEE Trans. Pattern Anal. Mach. Intell*. **13**, 1991, 1115–1138.

109. G. Taubin. F. Cukierman, S. Sullivan, J. Ponce, and D. J. Kriegman, Parameterized families of polynomials for bounded algebraic curve and surface fitting, *IEEE Trans. Pattern Anal. Mach. Intell*. **16**, 1994, 287–303.

110. J.-P. Thirion, The extremal mesh and the understanding of 3D surfaces, *Int. J. Comput. Vision* **19**, 1996, 115–128.

111. G. Turk, Re-tiling polygonal surfaces, in *Proceedings of SIGGRAPH '92*, pp. 55–64, 1992.

112. G. Turk and M. Levoy, Zippered polygon meshes from range images, in *Proceedings of SIGGRAPH '94, Orlando, Florida, July 1994*, pp. 311–318.

113. M. Turk and A. Pentland, Eigenfaces for recognition, *J. Cognitive Neurosci*. **3**, 1991, 71–86.

114. M. Turk and A. Pentland, Face recognition using eigenfaces, in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, pp. 586–591, 1991.

115. W. Welch and A. Witkin, Free-form shape design using triangulated surfaces, in *Proceedings of SIGGRAPH '94*, pp. 247–256, 1994.

116. D. Zhang and M. Hebert, Multi-scale classification of 3-D objects, in *Proc. IEEE Conf. Computer Vision and Pattern Recognition, San Juan, Puerto Rico, June 1997*, pp. 864–869.

117. Z. Zhang, Iterative point matching for registration of free-form curves and surfaces, *Int. J. Comput. Vision* **13**, 1994, 119–152.