Chapter 24

RANGE DATA

This chapter discusses *range images* that store, instead of
brightness or color information, the depth at which the ray associated with each pixel first intersects the scene observed by a camera. In a sense, a range image is exactly the desired output of stereo, motion, or other shape-from vision modules. In this chapter, however, we will focus our attention on range images acquired by *active* sensors, that project some sort of light pattern on the scene, using it to avoid the difficult and costly problem of establishing correspondences and construct dense and accurate depth pictures. After a brief review of range sensing technology, this chapter will discuss image segmentation, multiple-image registration and three-dimensional model construction, and object recognition, and explore the aspects of these problems that are specific to the range data domain.

24.1 Active Range Sensors

Triangulation-based range finders date back to the early seventies (e.g., [Agin, 1972; Shirai, 1972]). They function along the same principles as passive stereo vision systems, one of the cameras being replaced by a source of controlled illumination (*structured light*) that avoids the correspondence problem mentioned in Chapter 13. For example, a laser and a pair of rotating mirrors may be used to sequentially scan a surface. In this case, as in conventional stereo, the position of the bright spot where the laser beam strikes the surface of interest is found as the intersection of the beam with the projection ray joining the spot to its image. Contrary to the stereo case, however, the laser spot can normally be identified without difficulty since it is in general much brighter than the other scene points (in particular when a filter tuned to the laser wavelength is inserted in front of the camera), altogether avoiding the correspondence problem.

Alternatively, the laser beam can be transformed by a cylindrical lens into a plane of light (Figure 24.1). This simplifies the mechanical design of the range finder since it only requires one rotating mirror. More importantly, perhaps, it shortens the time required to acquire a range image since a laser stripe –the equivalent of a whole image column– can be acquired at each frame. It should be noted that this setup does not introduce matching ambiguities since the spot associated with each pixel can be retrieved as the (unique) intersection of the corresponding projection ray with the plane of light.

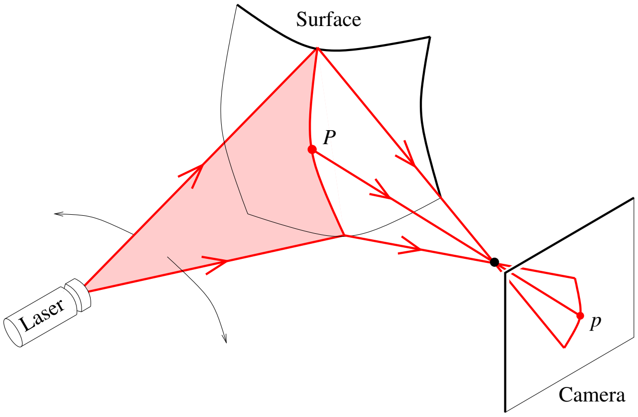

Figure 24.1. A range sensor using a plane of light to scan the surface of an object.

Variants of these two techniques include using multiple cameras to improve measurement accuracy and exploiting (possibly time-coded) two-dimensional light patterns to improve data acquisition speed. The main drawbacks of the active triangulation technology are relatively low acquisition speed, missing data at points where the laser spot is hidden from the camera by the object itself, and missing or erroneous data due to specularities. The latter difficulty is actually common to all active ranging techniques: a purely specular surface will not reflect any light in the direction of the camera unless it happens to lie in the corresponding mirror direction. Worse, the reflected beam may induce secondary reflections giving false depth measurements. Additional difficulties include keeping the laser stripe in focus during the entire scanning procedure, and the loss of accuracy inherent in all triangulation techniques as depth increases.

Several triangulation-based scanners are commercially available today. Figure 24.2 shows an example obtained using the Minolta VIVID range finder, that can acquire a 200×200 range image together with a registered 400×400 color image in 0.6s, within an operating range of 0.6 to 2.5m.

The second main approach to active ranging involves a signal transmitter, a receiver, and electronics for measuring the time of flight of the signal during its round trip from the range sensor to the surface of interest. This is the principle used in the ultrasound domain by the Polaroid range finder, commonly used in autofocus cameras from that brand and in mobile robots, despite the fact that the ultrasound wavelength band is particularly susceptible to false targets due to specular reflections. Time-of-flight laser range finders are normally equipped with a scanning mechanism, and the transmitter and receiver are often coaxial, eliminating the problem of missing data common in triangulation approaches. There are three main classes of time-of-flight laser range sensors: *pulse time delay* tech-

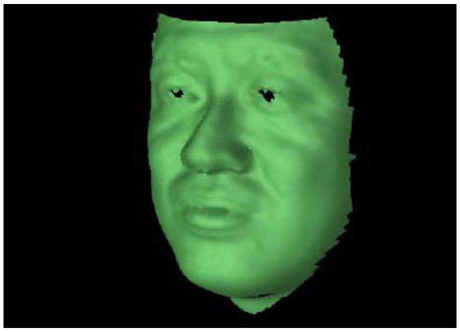

Figure 24.2. Range data captured by the Minolta VIVID scanner. As in most of the figures in this chapter, the mesh of $(x, y, z(x, y))$ points associated with the image is shown in perspective. Reprinted from [Hebert, 2000], Figure 6.

nology directly measures the time of flight of a laser pulse; *AM phase-shift* range finders measure the phase difference between the beam emitted by an amplitude-modulated laser and the reflected beam, a quantity proportional to the time of flight; finally, *FM beat* sensors measure the frequency shift (or *beat frequency*) between a frequency-modulated laser beam and its reflection, another quantity proportional to the round-trip flight time.

Time-of-flight range finders face the same problems as any other active sensors when imaging specular surfaces. They can be relatively slow due to long integration time at the receiver end. The speed of pulse-time delay sensors is also limited by the minimum resolvable interval between two pulses. AM phase-shift range finders suffer from inherent ambiguities since depth differences corresponding to phase shifts that are multiples of 2π cannot be resolved. This is a relatively minor problem since absolute range can normally be recovered by exploiting spatial coherence in the image, i.e., starting from the image points closest to the sensor, absolute depth can be propagated from one ambiguity interval to the next by a simple region-growing procedure. Compared to triangulation-based systems, time-of-flight sensors have the advantage of offering a greater operating range (up to tens of meters), which is very valuable in outdoor robotic navigation tasks.

Figure 24.3 shows range data acquired by a high-end AM phase-shift scanner [Hancock *et al.*, 1998], capable of acquiring 150,000 samples per second at a maximum range of 57m with an accuracy of 34mm.

As noted earlier, many range finders are now available commercially for a wide range of prices and applications. New technologies also continue to emerge, including range sensors equipped with acoustico-optical scanning systems and capable of extremely high image acquisition rates, and range cameras that eliminate scanning altogether, using instead a large array of receivers to analyze a laser pulse covering the entire field of view. Figure 24.4 shows an example of the latter technology, with images acquired by the Zcam range camera from 3DVSystems, which records full-frame registered range and color images at 30Hz with a depth resolution of

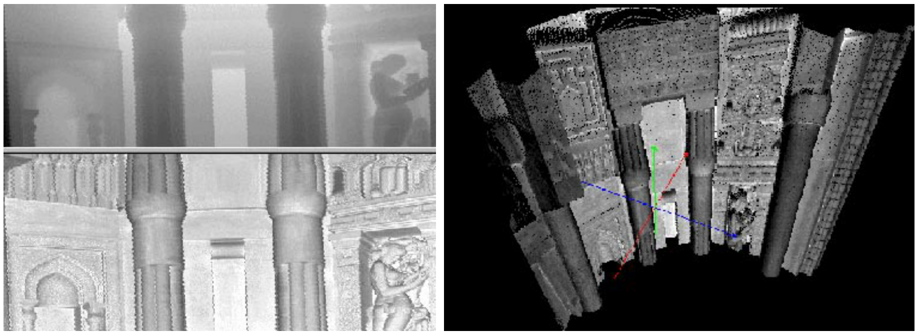

Figure 24.3. Range data captured by the AM phase shift range finder described in [Hancock *et al.*, 1998]: (left) range and intensity images; (right) perspective plot of the range data. Reprinted from [Hebert, 2000], Figure 5.

up to 10bits. To learn more, the interested reader should consult [Hebert, 2000] for an excellent discussion of current range finder technology, including a dozen of representative commercial products.

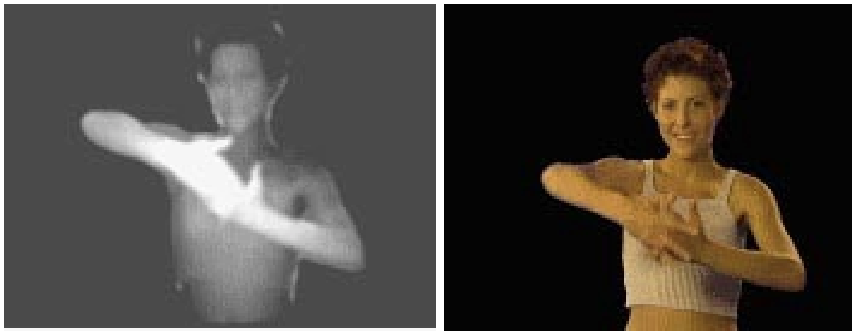

Figure 24.4. Range and color images captured by the Zcam range camera from 3DVSystems. Reprinted from [Hebert, 2000], Figure 10.

24.2 Range Data Segmentation

This section adapts some of the edge detection and segmentation methods introduced in Chapters **??** and **??** to the specific case of range images. As will be shown in the rest of this section, the fact that surface geometry is readily available greatly simplifies the segmentation process, mainly because this provides objective, physically-meaningful criteria for finding surface discontinuities and merging contiguous patches with a similar shape. But let us start by introducing some elementary notions of *analytical* differential geometry, which will turn out to form the basis for the approach to edge detection in range images discussed in this section.

Technique: Analytical Differential Geometry

Here we revisit the notions of differential geometry introduced in Chapter 5 in an analytical setting. Specifically, we consider a *parametric surface* defined as the smooth (i.e., indefinitely differentiable) mapping $\boldsymbol{x} : U \subset \mathbb{R}^2 \to \mathbb{R}^3$ that associates with any couple (u, v) in the open subset U of $\mathbb{R}^2$ the coordinate vector $\boldsymbol{x}(u, v)$ of a point in some fixed coordinate system. To ensure that the tangent plane is everywhere well defined, we will assume that the partial derivatives $\boldsymbol{x}_u \overset{\text{def}}{=} \partial \boldsymbol{x}/\partial u$ and $\boldsymbol{x}_v \overset{\text{def}}{=} \partial \boldsymbol{x}/\partial v$ are linearly independent. Indeed, let $\boldsymbol{\alpha} : I \subset \mathbb{R} \to U$ denote a smooth planar curve, with $\boldsymbol{\alpha}(t) = (u(t), v(t))$, then $\boldsymbol{\beta} \overset{\text{def}}{=} \boldsymbol{x} \circ \boldsymbol{\alpha}$ is a parameterized space curve lying on the surface. According to the chain rule, a tangent vector to $\boldsymbol{\beta}$ at the point $\boldsymbol{\beta}(t)$ is $u'(t)\boldsymbol{x}_u + v'(t)\boldsymbol{x}_v$, and it follows that the plane tangent to the surface in $\boldsymbol{x}(u, v)$ is parallel to the vector plane spanned by the vectors $\boldsymbol{x}_u$ and $\boldsymbol{x}_v$. The (unit) surface normal is thus

$$\boldsymbol{N} = \frac{1}{|\boldsymbol{x}_u \times \boldsymbol{x}_v|}(\boldsymbol{x}_u \times \boldsymbol{x}_v).$$

Let us consider a vector $\boldsymbol{t} = u'\boldsymbol{x}_u + v'\boldsymbol{x}_v$ in the tangent plane at the point $\boldsymbol{x}$. It is easy to show that the second fundamental form is given by

$$\mathrm{II}(\boldsymbol{t}, \boldsymbol{t}) = \boldsymbol{t} \cdot d\boldsymbol{N}(\boldsymbol{t}) = eu'^2 + 2fu'v' + gv'^2,$$

where[1]

$$e = -\boldsymbol{N} \cdot \boldsymbol{x}_{uu}, f = -\boldsymbol{N} \cdot \boldsymbol{x}_{uv}, g = -\boldsymbol{N} \cdot \boldsymbol{x}_{vv}.$$

Now if we define the *first fundamental form* as the bilinear form that associates with two vectors in the tangent plane their dot product, i.e.,

$$\mathrm{I}(\boldsymbol{u}, \boldsymbol{v}) \overset{\text{def}}{=} \boldsymbol{u} \cdot \boldsymbol{v},$$

then we have

$$\mathrm{I}(\boldsymbol{t}, \boldsymbol{t}) = |\boldsymbol{t}|^2 = Eu'^2 + 2Du'v' + Gv'^2,$$

where

$$E = \boldsymbol{x}_u \cdot \boldsymbol{x}_u, F = \boldsymbol{x}_u \cdot \boldsymbol{x}_v, G = \boldsymbol{x}_v \cdot \boldsymbol{x}_v.$$

It follows immediately that the normal curvature in the direction $\boldsymbol{t}$ is given by

$$\kappa_{\boldsymbol{t}} = \frac{\mathrm{II}(\boldsymbol{t}, \boldsymbol{t})}{\mathrm{I}(\boldsymbol{t}, \boldsymbol{t})} = \frac{eu'^2 + 2fu'v' + gv'^2}{Eu'^2 + 2Du'v' + Gv'^2}.$$

Likewise, it is easily shown that the matrix associated with the differential of the Gauss map *in the basis* $(\boldsymbol{x}_u, \boldsymbol{x}_v)$ of the tangent plane is

$$d\boldsymbol{N}(\boldsymbol{t}) = \begin{pmatrix} e & f \\ f & g \end{pmatrix} \begin{pmatrix} E & F \\ F & G \end{pmatrix}^{-1};$$

thus, since the Gaussian curvature is equal to the determinant of the operator $d\boldsymbol{N}$, it is given by

$$K = \frac{eg - f^2}{EG - F^2}.$$

[1] The definition of e, f and g is in keeping with the orientation conventions defined in Chapter 5. These coefficients are usually defined with opposite signs (e.g. [do Carmo, 1976; Struik, 1988]).

Asymptotic and principal directions are also easily found by using this parameterization: since an asymptotic direction verifies $\mathrm{II}(\boldsymbol{t},\boldsymbol{t})=0$, the corresponding values of u' and v' are the (homogeneous) solutions of $eu'^2+2fu'v'+gv'^2=0$. The principal directions, on the other hand, can be shown to verify

$$\begin{vmatrix} v'^2 & -u'v' & u'^2 \\ E & F & G \\ e & f & g \end{vmatrix}=0. \tag{24.2.1}$$

Example 1. An important example of parametric surface is provided by *Monge patches*: consider the surface $\boldsymbol{x}(u,v)=(u,v,h(u,v))$. In this case we have

$$\begin{cases} \boldsymbol{N}=\dfrac{1}{(1+h_u^2+h_v^2)^{1/2}}\begin{pmatrix}-h_u\\-h_v\\1\end{pmatrix}, \\ E=1+h_u^2, F=h_uh_v, G=1+h_v^2, \\ e=-\dfrac{h_{uu}}{(1+h_u^2+h_v^2)^{1/2}}, f=-\dfrac{h_{uv}}{(1+h_u^2+h_v^2)^{1/2}}, g=-\dfrac{h_{vv}}{(1+h_u^2+h_v^2)^{1/2}}, \end{cases}$$

and the Gaussian curvature has a simple form:

$$K=\frac{h_{uu}h_{vv}-h_{uv}^2}{(1+h_u^2+h_v^2)^2}.$$

Example 2. Another fundamental example is provided by the local parameterization of a surface in the coordinate system formed by its principal directions. This is of course a special case of a Monge patch. Writing that the origin of the coordinate system lies in the tangent plane immediately yields $h(0,0)=h_u(0,0)=h_v(0,0)=0$. As expected, the normal is simply $\boldsymbol{N}=(0,0,1)^T$ at the origin, and the first fundamental form is the identity there.

As shown in the exercises, it follows easily from (24.2.1) that a necessary and sufficient condition for the coordinate curves of a parameterized surface to be principal directions is that $f=F=0$ (this implies, for example, that the lines of curvature of a surface of revolution are its meridians and parallels). In our context we already know that $F=0$ and this condition reduces to $h_{uv}(0,0)=0$. The principal curvatures in this case are simply $\kappa_1=e/E=-h_{uu}(0,0)$ and $\kappa_2=g/G=-h_{vv}(0,0)$.

In particular, we can write a Taylor expansion of the height function in the neighborhood of $(0,0)$ as

$$h(u,v)=h(0,0)+uh_u(0,0)+vh_v(0,0)+\frac{1}{2}(u,v)\begin{pmatrix}h_{uu}(0,0) & h_{uv}(0,0)\\ h_{uv}(0,0) & h_{vv}(0,0)\end{pmatrix}\begin{pmatrix}u\\v\end{pmatrix}+\varepsilon(u^2+v^2)^{3/2},$$

which shows that the best second-order approximation to the surface in this neighborhood is the paraboloid defined by

$$h(u,v)=-\frac{1}{2}(\kappa_1u^2+\kappa_2v^2),$$

i.e., the expression already encountered in Chapter 5.

24.2.1 Finding Step and Roof Edges in Range Images

This section presents a method for finding instances of various types of edge models in range images [Ponce and Brady, 1987]. This technique combines tools from analytical differential geometry and scale-space image analysis to detect and locate depth and orientation discontinuities in range data. Figure 24.5 shows a 128 × 128 range image of a motor oil bottle that will serve to illustrate the concepts introduced in this section. This picture was acquired using the INRIA range finder [Boissonnat and Germain, 1981], with a depth accuracy of about 0.5mm.

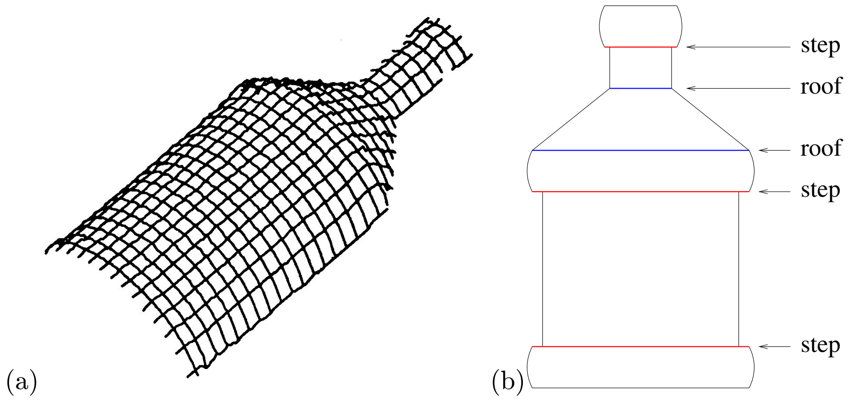

Figure 24.5. An oil bottle: (a) a range image of the bottle and (b) a sketch of its depth and orientation discontinuities.

The surface of the oil bottle presents two types of surface discontinuities: *steps*, where the actual depth is discontinuous, and *roofs*, where the depth is continuous but the orientation changes abruptly. As shown in the next section, it is possible to characterize the behavior of analytical models of step and roof edges under Gaussian smoothing and to show that they respectively give rise to parabolic points and extrema of the dominant principal curvature in the corresponding principal direction. This is the basis for the multi-scale edge detection scheme outlined in Algorithm 24.1 below.

Edge Models

In the neighborhood of a discontinuity, the shape of a surface changes much faster in the direction of the discontinuity than in the orthogonal direction. Accordingly, we will assume in the rest of this section that the direction of the discontinuity is one of the principal directions, with the corresponding (dominant) principal curvature changing rapidly in this direction, while the other one remains roughly equal to zero. This will allow us to limit our attention to *cylindrical* models of surface discontinuities, i.e., models of the form $z(x, y) = h(x)$. These models are of course only intended to be valid in the neighborhood of an edge, with the direction of the

1. Smooth the range image with Gaussian distributions at a set of scales σ_i ($i = 1, \ldots, 4$). Compute the principal directions and curvatures at each point of the smoothed images $z_{\sigma_i}(x, y)$.
2. Mark in each smoothed image $z_{\sigma_i}(x, y)$ the zero-crossings of the Gaussian curvature and the extrema of the dominant principal curvature in the corresponding principal direction.
3. Use the analytical step and roof models to match the features found across scales and output the points lying on these surface discontinuities.

Algorithm 24.1: *The model-based edge-detection algorithm of Ponce and Brady [1987].*

x, z plane being aligned with the corresponding dominant principal direction.

In particular, a step edge can be modeled by two sloped half-planes separated by a vertical gap, with normals in the $x - z$ plane. This model is cylindrical and it is sufficient to study its univariate formulation (Figure 24.6(left)), whose equation is

$$z = \begin{cases} k_1 x + c & \text{when} \quad x < 0, \\ k_2 x + c + h & \text{when} \quad x > 0. \end{cases} \tag{24.2.2}$$

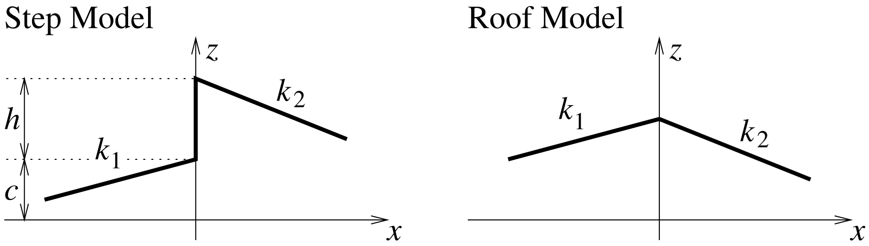

Figure 24.6. Edge models: a step consists of two half-planes separated by a distance h at the origin, and a roof consists of two half-planes meeting at the origin with different slopes. After [Ponce and Brady, 1987, Figure 4].

In this expression, c and h are constants, h measuring the size of the gap and k_1 and k_2 the slopes of the two half-planes. Introducing the new constants $k = (k_1 + k_2)/2$ and $\delta = k_2 - k_1$, it is easy to show (see exercises) that convolving the z function with the second derivative of a Gaussian yields

$$z''_\sigma \overset{\text{def}}{=} \frac{\partial^2}{\partial \sigma^2} G_\sigma * z = \frac{1}{\sigma\sqrt{2\pi}}(\delta - \frac{hx}{\sigma^2}) \exp(-\frac{x^2}{2\sigma^2}). \tag{24.2.3}$$

In particular, the corresponding curvature κ_σ vanishes in $x_\sigma = \sigma^2 \delta / h$. This point is only at the origin when $k_1 = k_2$ and its position is a quadratic function of σ otherwise. This suggests identifying step edges with zero-crossings of one of the principal curvatures (or equivalently of the Gaussian curvature), whose position changes with scale. To characterize qualitatively the behavior of these features as a function of σ, let us also note that since $z''_\sigma = 0$ in x_σ, we have

$$\frac{\kappa''_\sigma}{\kappa'_\sigma}(x_\sigma) = \frac{z''''_\sigma}{z''_\sigma}(x_\sigma) = -2\frac{\delta}{\sigma};$$

in other words, the ratio of the second and first derivatives of the curvature is independent of σ.

An analytical model for roof edges is obtained by taking $h = 0$ and $\delta \neq 0$ in the step model (Figure 24.6(right)). In this case, it is easy to show (see exercises) that

$$\kappa_\sigma = \frac{1}{\sigma\sqrt{2\pi}} \frac{\delta \exp(-\frac{x^2}{2\sigma^2})}{\left[1 + \left(k + \frac{\delta}{\sqrt{2\pi}} \int_0^{x/\sigma} \exp(-\frac{u^2}{2}) du\right)^2\right]^{3/2}}. \tag{24.2.4}$$

It follows that, when $x_2 = \lambda x_1$ and $\sigma_2 = \lambda\sigma_1$, we must have $\kappa_{\sigma_2}(x_2) = \kappa_{\sigma_1}(x_1)/\lambda$. In turn, the maximum value of $|\kappa_\sigma|$ must be inversely proportional to σ, and it is reached at a point whose distance from the origin is proportional to σ. This maximum tends toward infinity as σ tends toward zero, indicating that roofs can be found as local curvature extrema. In actual range images, these extrema should be sought in the direction of the dominant principal direction, in keeping with our assumptions about local shape changes in the vicinity of surface edges.

Computing the Principal Curvatures and Directions

According to the models derived in the previous section, instances of step and roof edges can be found as zero crossings of the Gaussian curvature and extrema of the dominant principal curvature in the corresponding direction. Computing these differential quantities requires estimating the first and second partial derivatives of the depth function at each point of a range image. This can be done, as in Chapter **??**, by convolving the images with the derivatives of a Gaussian distribution. However, range images are different from usual pictures: for example, the pixel values in a photograph are usually assumed to be piecewise constant in the neighborhood of step edges,[2] which is justified for Lambertian objects since the shape of a surface is, to first order, piecewise-planar near an edge, with a piecewise-planar intensity in that case. On the other hand, piecewise-constant (local) models of range images are of course unsatisfactory. Likewise, the maximum values of contrast along the significant edges of a photograph are usually assumed to have roughly the same

[2] This corresponds to taking $k_1 = k_2 = 0$ in the model given in the previous section; note that in that case zero crossings do not move as scale changes.

magnitude. In range images, however, there are two different types of step edges: the large depth discontinuities that separate solid objects from each other and from their background, and the much smaller gaps that usually separate patches of the same surface.

The edge detection scheme discussed in this section is aimed at the latter class of discontinuities. Blindly applying Gaussian smoothing across object boundaries will introduce radical shape changes that may overwhelm the surface details we are interested in (Figure 24.7(top and middle)).

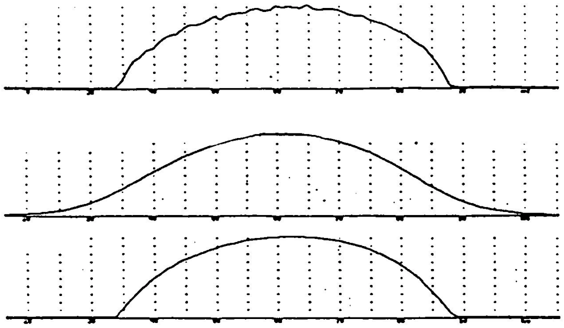

Figure 24.7. Smoothing a range image. Top: a slice of the range image shown in Figure 24.5. The background has been thresholded away. Middle: result of Gaussian smoothing. Bottom: smoothing using computational molecules. Reprinted from [Ponce and Brady, 1987], Figure 14.

This suggests finding the major depth discontinuities first (thresholding will suffice in many cases), then somehow restricting the smoothing process to the surface patches enclosed by these boundaries. This can be achieved by convolving the range image with *computational molecules* [Terzopoulos, 1984], i.e., linear templates that, added together, form a 3 × 3 averaging mask, e.g.,

$$\begin{array}{ccc} \boxed{1} & & \\ & \boxed{2} & \\ & & \boxed{1} \end{array} + \begin{array}{|c|c|c|}\hline 2 & 4 & 2 \\ \hline \end{array} + \begin{array}{|c|}\hline 2 \\ \hline 4 \\ \hline 2 \\ \hline \end{array} + \begin{array}{ccc} & & \boxed{1} \\ & \boxed{2} & \\ \boxed{1} & & \end{array} = \begin{array}{|c|c|c|}\hline 1 & 2 & 1 \\ \hline 2 & 12 & 2 \\ \hline 1 & 2 & 1 \\ \hline \end{array}.$$

Repeatedly convolving the image with the 3 × 3 mask (normalized so its weights add to one) yields, according to the central limit theorem, a very good approximation of Gaussian smoothing with a mask whose σ value is proportional to $\sqrt{n}$ after n iterations. To avoid smoothing across discontinuities, the molecules crossing these discontinuities are not used, while the remaining ones are once again normalized so the total sum of the weights is equal to one. The effect is shown in Figure 24.7(bottom).

After the surface has been smoothed, the derivatives of the height function can be computed via finite differences. The gradient of the height function is computed by convolving the smoothed image with the masks:

$$\frac{\partial}{\partial x} = \frac{1}{6}\begin{array}{|c|c|c|}\hline -1 & 0 & 1 \\ \hline -1 & 0 & 1 \\ \hline -1 & 0 & 1 \\ \hline \end{array} \quad \text{and} \quad \frac{\partial}{\partial y} = \frac{1}{6}\begin{array}{|c|c|c|}\hline 1 & 1 & 1 \\ \hline 0 & 0 & 0 \\ \hline -1 & -1 & -1 \\ \hline \end{array},$$

and the Hessian is computed by convolving the smoothed image with the masks

$$\frac{\partial^2}{\partial x^2} = \frac{1}{3}\begin{array}{|c|c|c|}\hline 1 & -2 & 1 \\ \hline 1 & -2 & 1 \\ \hline 1 & -2 & 1 \\ \hline \end{array}, \quad \frac{\partial^2}{\partial x \partial y} = \frac{1}{4}\begin{array}{|c|c|c|}\hline -1 & 0 & 1 \\ \hline 0 & 0 & 0 \\ \hline 1 & 0 & -1 \\ \hline \end{array} \quad \text{and} \quad \frac{\partial^2}{\partial y^2} = \frac{1}{3}\begin{array}{|c|c|c|}\hline 1 & 1 & 1 \\ \hline -2 & -2 & -2 \\ \hline 1 & 1 & 1 \\ \hline \end{array}.$$

Once the derivatives are known, the principal directions and curvatures are easily computed. Figure 24.8 shows the two sets of principal directions found for the oil bottle after 20 iterations of the molecules. As expected, they lie along the meridians and parallels of this surface of revolution.

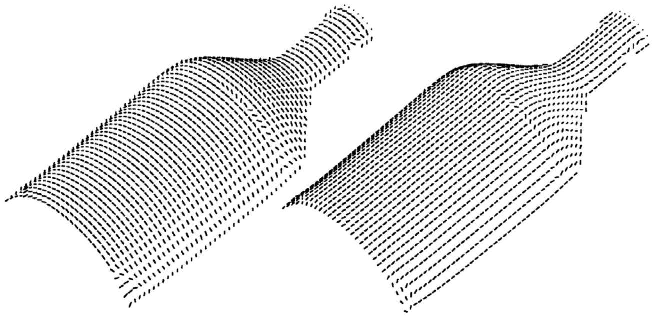

Figure 24.8. The two principal direction fields for the oil bottle. Reprinted from [Brady *et al.*, 1985], Figure 18.

Matching Features Across Scales

Given the principal curvatures and directions, parabolic points can be detected as (non-directional) zero-crossings of the Gaussian curvature, while local extrema of the dominant curvature along the corresponding principal direction can be found using the non-maximum suppresion techniques discussed in Chapter **??**. Figure 24.9(a) shows the features found after 20, 40, 60, and 80 iterations of the molecule-based smoothing operator. Although there is a considerable amount of noise at fine resolutions (e.g., after 20 iterations only), the situation improves as smoothing proceeds. Features due to noise can also be eliminated, at least in part, via thresholding of the zero-crossing slope for parabolic points, and of the curvature magnitude for extrema of principal curvatures (Figure 24.9(b)).

Nonetheless, experiments show that smoothing and thresholding are not sufficient to eliminate as much as possible all irrelevant features. In particular, as

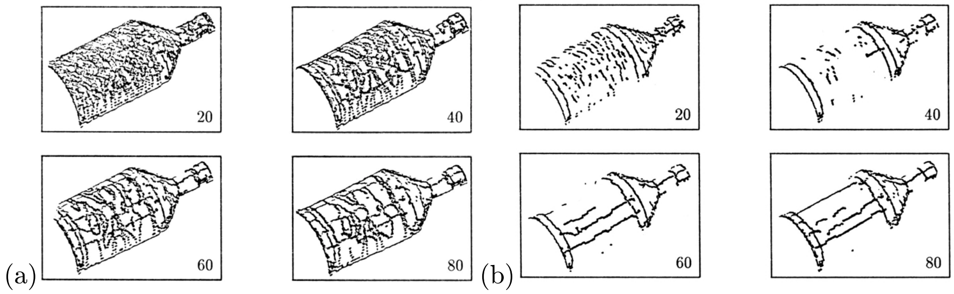

Figure 24.9. Features found at various scales (a) before and (b) after thresholding. Note that the thresholds in (b) have been chosen empirically to eliminate most false features while retaining those corresponding to true surface discontinuities. Still, artefacts such as the extrema of curvature parallel to the axis of the bottle subsist. Reprinted from [Ponce and Brady, 1987], Figure 12.

illustrated by Figure 24.9, curvature extrema parallel to the axis of the oil bottle show up more and more clearly as smoothing proceeds. These are due to the fact that points near the occluding boundary of the bottle do not get smoothed as much by the computational molecules as points closer to its center, giving rise to artificial curvature extrema.

A multi-scale approach to edge detection solves this problem. Features are tracked from coarse to fine scales, all features at a given scale not having an ancestor at a coarser one being eliminated. The evolution of the principal curvatures and their derivatives is also monitored. Surviving parabolic features such that the ratio $\kappa''_\sigma/\kappa'_\sigma$ remains (roughly) constant across scales are output as step edge points, while directional extrema of the dominant curvature such that $\sigma\kappa_\sigma$ remains (roughly) constant are output as roof points. Finally, since, for both our models, the distance between the true discontinuity and the corresponding zero crossing or extremum increases with scale, the finest scale is used for edge localization. Figure 24.10 shows the results of applying this strategy to the oil bottle and a human face mask.

24.2.2 Segmenting Range Images into Planar Regions

We saw in the last section that edge detection is implemented by quite different processes in photographs and range data. The situation is similar for image segmentation into regions. In particular, meaningful segmentation criteria are elusive in the intensity domain because pixel brightness is only a cue to physical properties such as shape or reflectance. In the range domain however, geometric information is directly available, making it possible to use, say, the average distance between a set of surface points and the plane best fitting them as an effective segmentation criterion. The region growing technique of Faugeras and Hebert [1986] is a good example of this approach. This algorithm iteratively merges planar patches by maintaining a graph whose nodes are the patches and arcs associated with their

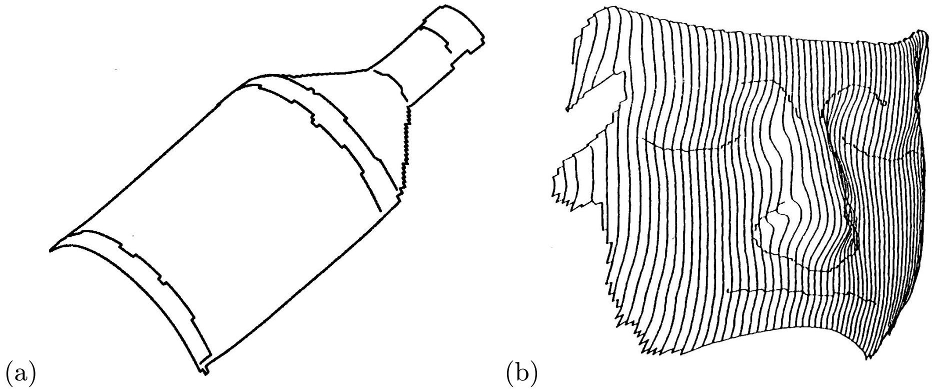

Figure 24.10. Edge detection results: (a) the three step edges and two roof discontinuities of the oil bottle have been correctly identified; (b) the eye, mouth, nose and brow boundaries of a mask have been found as roof edges. Reprinted from [Ponce and Brady, 1987], Figures 16 and 22.

common boundary link adjacent patches. Each arc is assigned a cost corresponding to the average error between the points of the two patches and the plane best fitting these points. The best arc is always selected, and the corresponding patches are merged. Note that the remaining arcs associated with these patches must be deleted while new arcs linking the new patch to its neighbors are introduced. The situation is illustrated by Figure 24.11.

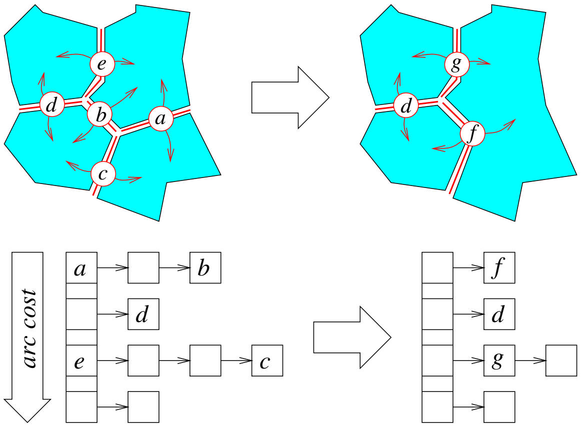

Figure 24.11. This diagram illustrates one iteration of the region growing process during which the two patches incident to the minimum-cost arc labelled a are merged. The heap shown in the bottom part of the figure is updated as well: the arcs a, b, c and e are deleted, and two new arcs f and g are created and inserted in the heap.

The graph structure is initialized by using a triangulation of the range data, and it is efficiently updated by maintaining a heap of active arcs. The triangulation can either be constructed directly from a range image (by splitting the quadrilaterals associated with the pixels along one of their diagonals), or from a global surface model constructed from multiple images as described in the next section. The heap storing the active arcs can be represented, for example, by an array of buckets indexed by increasing costs, which supports fast insertion and deletion (Figure 24.11(bottom)). Figure 24.12 shows an example, where the complex shape of an automobile part is approximated by 60 planar patches.

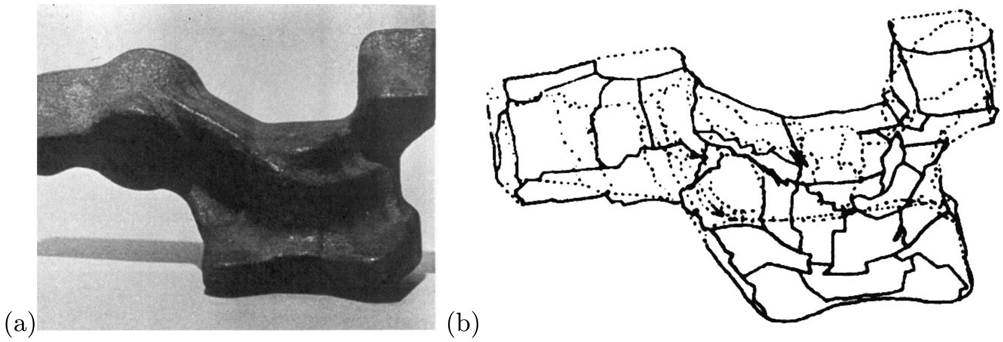

Figure 24.12. The Renault part: (a) photo of the part and (b) its model. Reprinted from [Faugeras and Hebert, 1986], Figures 1 and 6.

24.3 Range Image Registration and Model Construction

Geometric models of real objects are useful in manufacturing, e.g., for process and assembly planning or inspection. Closer to the theme of this book, they are also key components of many object recognition systems, and are more and more in demand in the entertainment industry, as synthetic pictures of real objects now routinely appear in feature films and video games (we will come back to this issue in much greater detail in Chapter 26). Range images are an excellent source of data for constructing accurate geometric models of real objects, but a single picture will, at best, show half of the surface of a given solid, and the construction of complete object models requires the integration of multiple range images. This section addresses the dual problems of registering multiple images in the same coordinate system and fusing the three-dimensional data provided by these pictures into a single integrated surface model.

Before attacking these two problems, let us introduce quaternions, that will provide us with linear methods for estimating rigid transformations from point and plane correspondences in both the registration context of this section and the recognition context of the next one.

Technique: Quaternions

Quaternions were invented by Hamilton [1844]. Like complex numbers in the plane, they can be used to represent rotations in space in a very convenient manner. A quaternion q is defined by its *real part*, a scalar a, and its *imaginary part*, a vector $\boldsymbol{\alpha}$ in $\mathbb{R}^3$, and it is usually denoted by $\mathsf{q} = a + \boldsymbol{\alpha}$. This is justified by the fact that real numbers can be identified with quaternions with a zero imaginary part, and vectors can be identified with quaternions with a zero real part, while addition between quaternions is defined by

$$(a + \boldsymbol{\alpha}) + (b + \boldsymbol{\beta}) \stackrel{\text{def}}{=} (a + b) + (\boldsymbol{\alpha} + \boldsymbol{\beta}).$$

The multiplication of a quaternion by a scalar is defined naturally by $\lambda(a + \boldsymbol{\alpha}) \stackrel{\text{def}}{=} \lambda a + \lambda\boldsymbol{\alpha}$, and these two operations give the set of all quaternions the structure of a four-dimensional vector space.

It is also possible to define a multiplication operation that associates with two quaternions the quaternion

$$(a + \boldsymbol{\alpha})(b + \boldsymbol{\beta}) \stackrel{\text{def}}{=} (ab - \boldsymbol{\alpha} \cdot \boldsymbol{\beta}) + (a\boldsymbol{\beta} + b\boldsymbol{\alpha} + \boldsymbol{\alpha} \times \boldsymbol{\beta}).$$

Quaternions, equipped with the operations of addition and multiplication as defined above, form a non-commutative field, whose zero and unit elements are respectively the scalars 0 and 1.

The *conjugate* of the quaternion $\mathsf{q} = a + \boldsymbol{\alpha}$ is the quaternion $\bar{\mathsf{q}} \stackrel{\text{def}}{=} a - \boldsymbol{\alpha}$ with opposite imaginary part. The squared norm of a quaternion is defined by

$$|\mathsf{q}|^2 \stackrel{\text{def}}{=} \mathsf{q}\bar{\mathsf{q}} = \bar{\mathsf{q}}\mathsf{q} = a^2 + |\boldsymbol{\alpha}|^2,$$

and it is easily verified that $|\mathsf{q}\mathsf{q}'| = |\mathsf{q}||\mathsf{q}'|$ for any pair of quaternions q and q'.

Now, it can be shown that the quaternion

$$\mathsf{q} = \cos\frac{\theta}{2} + \sin\frac{\theta}{2}\boldsymbol{u}$$

represents the rotation $\mathcal{R}$ of angle θ about the *unit* vector $\boldsymbol{u}$ in the following sense: if $\boldsymbol{\alpha}$ is some vector in $\mathbb{R}^3$, then

$$\mathcal{R}\boldsymbol{\alpha} = \mathsf{q}\boldsymbol{\alpha}\bar{\mathsf{q}}. \tag{24.3.1}$$

Note that $|\mathsf{q}| = 1$ and that $-\mathsf{q}$ also represents the rotation $\mathcal{R}$. Reciprocally, the rotation matrix $\mathcal{R}$ associated with a given unit quaternion $\mathsf{q} = a + \boldsymbol{\alpha}$ with $\boldsymbol{\alpha} = (b, c, d)^T$ is

$$\mathcal{R} = \begin{pmatrix} a^2 + b^2 - c^2 - d^2 & 2(bc - ad) & 2(bd + ac) \\ 2(bc + ad) & a^2 - b^2 + c^2 - d^2 & 2(cd - ab) \\ 2(bd - ac) & 2(cd + ab) & a^2 - b^2 - c^2 + d^2 \end{pmatrix},$$

a fact easily deduced from (24.3.1). (Note that the four parameters a, b, c, d are not independent since they satisfy the constraint $a^2 + b^2 + c^2 + d^2 = 1$.)

Finally, if q_1 and q_2 are unit quaternions, and $\mathcal{R}_1$ and $\mathcal{R}_2$ are the corresponding rotation matrices, the quaternions $\mathsf{q}_1\mathsf{q}_2$ and $-\mathsf{q}_1\mathsf{q}_2$ are both representations of the rotation matrix $\mathcal{R}_1\mathcal{R}_2$.

24.3.1 Registering Range Images Using the Iterative Closest-Point Method

Besl and McKay [1992] have proposed an algorithm capable of registering two sets of three-dimensional points, i.e., of computing the rigid transformation that maps the first point set onto the second one. Their algorithm simply minimizes the average distance between the two point sets by iterating over the following steps: first establish correspondences between scene and model features by matching every scene point to the model point closest to it, then estimate the rigid transformation mapping the scene points onto their matches, and finally apply the computed displacement to the scene. The iterations stop when the change in mean distance between the matched points falls below some preset threshold. Pseudocode for this *iterated closest-point* (or *ICP*) algorithm is given below.

```
Function ICP(Model, Scene);
begin
E' $\leftarrow +\infty$;
(Rot, Trans) $\leftarrow$ Initialize-Registration(Scene, Model);
repeat
  E $\leftarrow$ E';
  Registered-Scene $\leftarrow$ Apply-Registration(Scene, Rot, Trans);
  Pairs $\leftarrow$ Return-Closest-Pairs(Registered-Scene, Model);
  (Rot, Trans, E') $\leftarrow$ Update-Registration(Scene, Model, Pairs, Rot, Trans);
  until $|E' - E| < \tau$;
return (Rot, Trans);
end.
```

Algorithm 24.2: *The iterative closest-point algorithm of Best and McKay [1992].*

The auxiliary function Initialize-Registration uses some global registration method, based on moments for example, to compute a rough initial estimate of the rigid transformation mapping the scene onto the model. The function Return-Closest-Pairs returns the indices (i, j) of the points in the registered scene and the model such that point number j is the closest to point number i. The function Update-Registration estimates the rigid transformation between selected pairs of points in the scene and the model, and the function Apply-Registration applies a rigid transformation to all the points in the scene.

It can be shown that Algorithm 24.2 always converges to a local minimum of the error E (this is intuitively clear since the registration stage decreases the average error at each iteration, while the closest point determination decreases the individual error as well). This does not guarantee of course, convergence to a global minimum, and a reasonable guess for the rigid transformation sought by the algorithm must be provided. A variety of methods are available for that purpose, including roughly sampling the set of all possible transformations, and using the moments of both the scene and model point sets to estimate the transformation.

Finding the Closest-Point Pairs

At every iteration of the algorithm, finding the closest point M in the model to a given (registered) scene point S takes (naively) $O(n)$ time, where n is the number of model points. In fact, various algorithms can be used to answer such a nearest-neighbor query in $\mathbb{R}^3$ in $O(\log n)$ time at the cost of additional preprocessing of the model, using for example *k-d trees* [Friedman *et al.*, 1977] (for which the logarithmic query time only holds on average) or more complex data structures. For example, the general randomized algorithm of [Clarkson, 1988] takes preprocessing time $O(n^{2+\varepsilon})$, where ε is an arbitrarily small positive number, and query time $O(\log n)$. The efficiency of repeated queries can also be improved by *caching* the results of previous computations. For example, Simon *et al.* [1994] store at each iteration of the ICP algorithm the k closest model points to each scene point (a typical value for k is 5). Since the incremental update of the rigid transformation is normally small, it is likely that the closest neighbor of a point after an iteration will be among its k closest neighbors from the previous one. It is in fact possible to determine efficiently and conclusively whether the closest point is in the cached set, see [Simon *et al.*, 1994] for details.

Estimating the Rigid Transformation

Under the rigid transformation defined by the rotation matrix $\mathcal{R}$ and the translation vector $\boldsymbol{t}$, a point $\boldsymbol{x}$ maps onto the point $\boldsymbol{x}' = \mathcal{R}\boldsymbol{x}+\boldsymbol{t}$. Thus, given n pairs of matching points $\boldsymbol{x}_i$ and $\boldsymbol{x}'_i$, with $i = 1, \ldots, n$, we seek the rotation matrix $\mathcal{R}$ and translation vector $\boldsymbol{t}$ minimizing the error

$$E = \sum_{i=1}^{n} |\boldsymbol{x}'_i - \mathcal{R}\boldsymbol{x}_i - \boldsymbol{t}|^2.$$

Let us first note that the value of $\boldsymbol{t}$ minimizing E must satisfy

$$0 = \frac{\partial E}{\partial \boldsymbol{t}} = -2\sum_{i=1}^{n}(\boldsymbol{x}'_i - \mathcal{R}\boldsymbol{x}_i - \boldsymbol{t}),$$

or

$$\boldsymbol{t} = \boldsymbol{x}'_0 - \mathcal{R}\boldsymbol{x}_0, \quad \text{where} \quad \boldsymbol{x}_0 \stackrel{\text{def}}{=} \frac{1}{n}\sum_{i=1}^{n}\boldsymbol{x}_i \quad \text{and} \quad \boldsymbol{x}'_0 \stackrel{\text{def}}{=} \frac{1}{n}\sum_{i=1}^{n}\boldsymbol{x}'_i \tag{24.3.2}$$

denote respectively the centroids of the two sets of points $\boldsymbol{x}_i$ and $\boldsymbol{x}'_i$.

Introducing the centered points $\boldsymbol{y}_i = \boldsymbol{x}_i - \boldsymbol{x}_0$ and $\boldsymbol{y}'_i = \boldsymbol{x}'_i - \boldsymbol{x}_0$ $(i = 1, \ldots, n)$ yields

$$E = \sum_{i=1}^{n} |\boldsymbol{y}'_i - \mathcal{R}\boldsymbol{y}_i|^2,$$

Quaternions can now be used to minimize E as follows: let q denote the quaternion associated with the matrix $\mathcal{R}$, we use the fact that $|\mathsf{q}|^2 = 1$ and the multiplicativity properties of the quaternion norm to write

$$E = \sum_{i=1}^{n} |\boldsymbol{y}'_i - \mathsf{q}\boldsymbol{y}_i\bar{\mathsf{q}}|^2 |\mathsf{q}|^2 = \sum_{i=1}^{n} |\boldsymbol{y}'_i\mathsf{q} - \mathsf{q}\boldsymbol{y}_i|^2.$$

As shown in the exercises, this allows us to rewrite the rotational error as $E = \mathsf{q}^T \mathcal{B}\mathsf{q}$, where $\mathcal{B} = \sum_{i=1}^{n} \mathcal{A}_i^T \mathcal{A}_i$, and

$$\mathcal{A}_i = \begin{pmatrix} 0 & \boldsymbol{y}_i^T - \boldsymbol{y}_i'^T \\ \boldsymbol{y}'_i - \boldsymbol{y}_i & [\boldsymbol{y}_i + \boldsymbol{y}'_i]_\times \end{pmatrix}.$$

Note that the matrix $\mathcal{A}_i$ is antisymmetric with (in general) rank 3, but that the matrix $\mathcal{B}$ will have, in the presence of noise, rank 4. As shown in Chapter 6, minimizing E under the constraint $|\mathsf{q}|^2 = 1$ is a (homogeneous) linear least-squares problem whose solution is the eigenvector of $\mathcal{B}$ associated with the smallest eigenvalue of this matrix. Once $\mathcal{R}$ is known, $\boldsymbol{t}$ is obtained from (24.3.2).

Results

Figure 24.13 shows an example, where two range images of an African mask are matched by the algorithm. The average distance between matches is 0.59mm for this 9cm object.

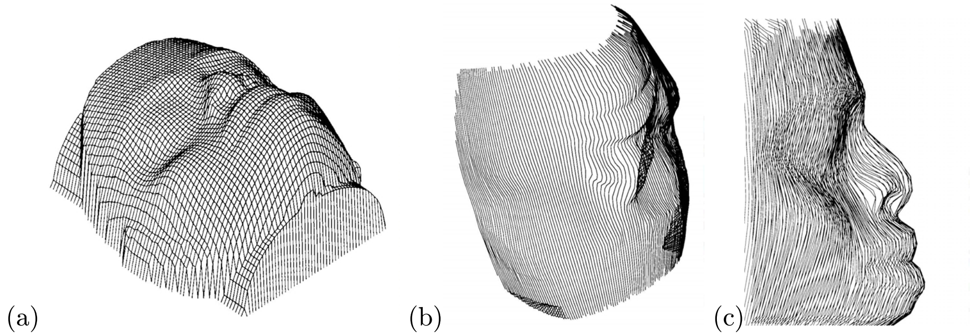

Figure 24.13. Registration results: (a) a range image serving as model for an African mask; (b) a (decimated) view of the model, serving as scene data; (c) a view of the two datasets overlaid after registration. Reprinted from [Besl and McKay, 1992], Figures 12–14.

This method is not limited to models consisting to clouds of three-dimensional points, but applies as well to any model that supports the construction of closest-point pairs. Figure 24.14 shows an example where a range image is matched to a spline model of the mask. In this case, the patch point closest to a scene point is retrieved by a simple optimization process, initialized at the center of the patch for example (see exercises).

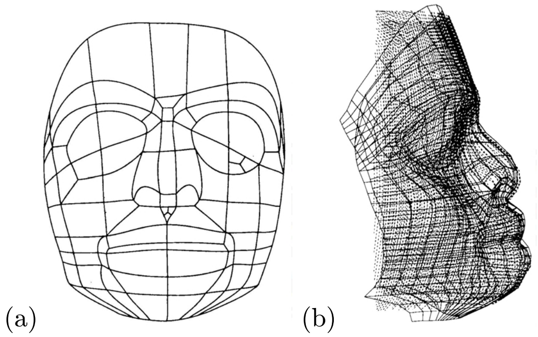

Figure 24.14. More registration results: (a) a parametric surface serving as model for the mask; (b) registration of this model and the range image shown in Figure 24.13. Reprinted from [Besl and McKay, 1992], Figures 15 and 16.

24.3.2 Fusing Multiple Range Images

Given a set of registered range images of a solid object, it is possible to construct an integrated surface model of this object. In the approach proposed by Curless and Levoy [1996], this model is constructed as the zero set S of a volumetric density function $D : \mathbb{R}^3 \rightarrow \mathbb{R}$, i.e., as the set of points (x, y, z) such that $D(x, y, z) = 0$. Like any other level set of a continuous density function, S is by construction guaranteed to be a closed, "watertight" surface, although it may have several connected components (Figure 24.15)

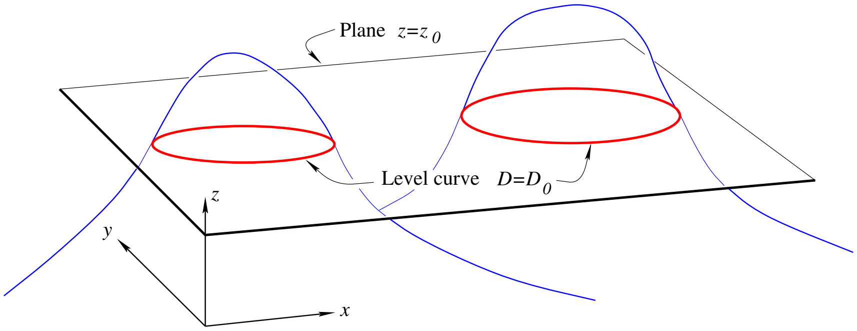

Figure 24.15. A 2D illustration of volumetric density functions and their level sets. In this case, the "volume" is of course the (x, y) plane, and the "surface" is a curve in this plane, with two connected components in the example shown here.

The trick, of course, is to construct an appropriate density function from registered range measurements. Curless and Levoy embed the corresponding surface fragments into a cubic grid, and assign to each cell of this grid, or *voxel*, a weighted sum of the signed distances between its center and the closest point on the surface intersecting it (Figure 24.16(left)). This averaged signed distance is the desired density function, and its zero set can be found using classical techniques, such as the *marching cubes* algorithm develop by Lorensen and Cline [1987] to extract isodensity surfaces from volumetric medical data.

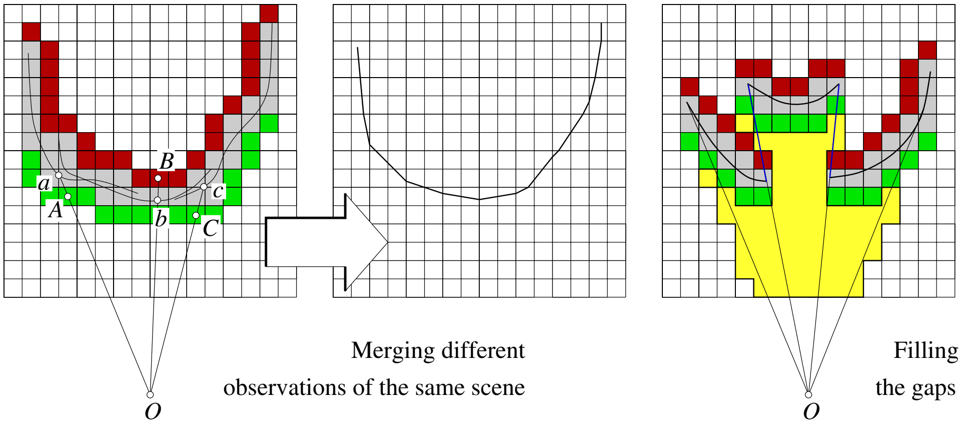

Figure 24.16. A 2D illustration of the Curless-Levoy method for fusing multiple range images. In the left part of the figure, three views observed by the same sensor located at the point O are merged by computing the zero set of a weighted average of the signed distances between voxel centers (e.g., points A, B and C) and surface points (e.g., a, b and c) along viewing rays. In general, distances to different sensors would be used instead. The light grey area in the right part of the figure is the set of voxels marked as empty in the gap-filling part of the procedure.

Missing surface fragments corresponding to unobserved parts of the scene are handled by initially marking all voxels as *unseen*, or equivalently assigning them a depth equal to some large positive value (standing for $+\infty$), then assigning as before to all voxels close to the measured surface patches the corresponding signed distance, and finally carving out (i.e., marking as *empty*, or having a large negative depth standing for $-\infty$) the voxels that lie between the observed surface patches and the sensor (Figure 24.16(right)).

Figure 24.17 shows an example of model built from multpiple range images of a Buddha statuette acquired with a Cyberware 3030 MS optical triangulation scanner, as well as a physical model constructed from the geometric one via stereolithography [Curless and Levoy, 1996].

24.4 Object Recognition

We now turn to actual object recognition from range images. The registration techniques introduced in the previous section will play a crucial role in the two algorithms discussed in this one.

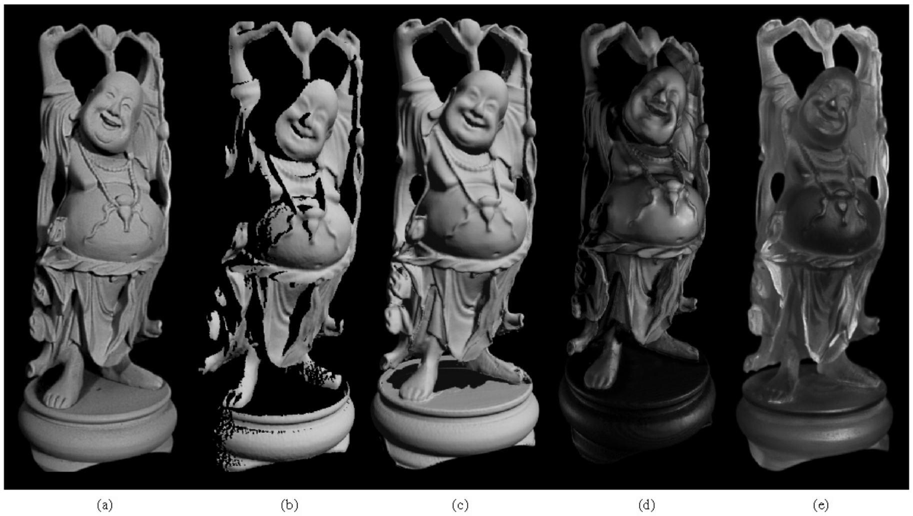

Figure 24.17. 3D Fax of a statuette of a Buddha. From left to right: photograph of the statuette; range image; integrated 3D model; model after hole filling; physical model obtained via stereolithography. Reprinted from [Curless and Levoy, 1996], Figure 10.

24.4.1 Matching Piecewise-Planar Surfaces Using Interpretation Trees

The recognition algorithm proposed by Faugeras and Hebert [1986] is a recursive procedure exploiting rigidity constraints to efficiently search an interpretation tree for the path(s) corresponding to the best sequence(s) of matches. The basic procedure is given in pseudocode in Algorithm 24.3 below. To correctly handle occlusions (and the fact that, as noted earlier, a range finder will "see", at best, one half of the object facing it), the algorithm must consider, at every stage of the search, the possibility that a model plane may not match any scene plane. This is done by always incorporating in the list of potential matches of a given plane a token "null" plane.

Selecting Potential Matches

The selection of potential matches for a given model plane is based on various criteria depending on the number of correspondences already established, with each new correspondence providing new geometric constraints and more stringent criteria. At the beginning of the search, we only know that a model plane with area A should only be matched to scene planes with a compatible area, i.e., in the range $[\alpha A, \beta A]$. Reasonable values for the two thresholds might be 0.5 and 1.1, which allows for some

```
Function Match(model, scene, pairs, rot, trans);
begin
bestpairs $\leftarrow$ nil; bestscore $\leftarrow$ 0;
for $\Pi$ in model do
   for $\Pi'$ in Potential-Matches(scene, pairs, $\Pi$, rot, trans) do
     rot $\leftarrow$ Update-Registration-2(pairs, $\Pi$, $\Pi'$, rot, trans);
     (score, newpairs) $\leftarrow$ Match(model$-\Pi$, scene$-\Pi'$, pairs$+(\Pi,\Pi')$, rot, trans);
     if score>bestscore then bestscore $\leftarrow$ score; bestpairs $\leftarrow$ newpairs endif;
     endfor;
   endfor;
return bestpairs;
end.
```

Algorithm 24.3: *The plane-matching algorithm of Faugeras and Hebert [1986].*

The recursive function Match returns the best set of matching plane pairs found by recursively visiting the interpretation tree. It is initially called with an empty list of pairs and nil values for the rotation and translation arguments rot and trans. The auxiliary function Potential-Matches returns the subset of the planes in the scene that are compatible with the model plane Π *and the current estimate of the rigid transformation mapping the model planes onto their scene matches (see text for details). The auxiliary function Update-Registration-2 uses the matched plane pairs to update the current estimate of the rigid transformation.*

discrepency between the unoccluded areas, and also affords a degree of occlusion up to 50%.

After the first correspondence has been established, it is still too early to try and estimate the rigid transformation mapping the model onto the scene, but it is clear that the angle between the normals to any matching planes should be (roughly) equal to the angle θ between the normals to the first pair of planes, say lie in the interval $[\theta - \varepsilon, \theta + \varepsilon]$. The normals to the corresponding planes lie in a band of the Gauss sphere, and they can be efficiently retrieved by discretizing this sphere and associating to each cell a bucket that stores the scene planes whose normal falls into it (Figure 24.18).

A second pairing is sufficient to complete determine the rotation separating the model from its instance in the scene: this is geometrically clear (and will be confirmed analytically in the next section) since a pair of matching vectors constrains the rotation axis to lie in the plane bisecting these vectors. Two pairs of matching planes determine the axis of rotation as the intersection of the corresponding bisecting planes, and the rotation angle is readily computed from either of the matches. Given the rotation and a third model plane, one can predict the orientation of the normal to its possible matches in the scene, which can be efficiently recovered using once again the discrete Gauss sphere mentioned before.

After three pairings have been found, the translation can also be estimated and

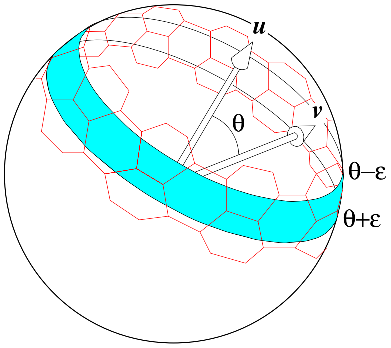

Figure 24.18. Finding all vectors $\boldsymbol{v}$ that make an angle in the $[\theta - \varepsilon, \theta + \varepsilon]$ range with a given vector $\boldsymbol{u}$. It should be noted that the unit sphere does not admit tesselations with an arbitrary level of detail by regular (spherical) polygons. The tesselation shown in the diagram is made of hexagons with unequal edge lengths. See, for example, [Horn, 1986, Chap. 16] for a discussion of this problem and various tesselation schemes.

used to predict the distance between the origin and any scene plane matching a fourth scene plane. The same is true for any further pairing.

Estimating the Rigid Transformation

Let us consider a plane Π defined by the equation $\boldsymbol{n} \cdot \boldsymbol{x} - d = 0$ in some fixed coordinate system. Here $\boldsymbol{n}$ denotes the unit normal to the plane and d its (signed) distance from the origin. Under the rigid transformation defined by the rotation matrix $\mathcal{R}$ and the translation vector $\boldsymbol{t}$, a point $\boldsymbol{x}$ maps onto the point $\boldsymbol{x}' = \mathcal{R}\boldsymbol{x} + \boldsymbol{t}$, and Π maps onto the plane Π' whose equation is $\boldsymbol{n}' \cdot \boldsymbol{x}' - d' = 0$, with

$$\begin{cases} \boldsymbol{n}' = \mathcal{R}\boldsymbol{n}, \\ d' = \boldsymbol{n}' \cdot \boldsymbol{t} + d. \end{cases}$$

Thus, estimating the rigid transformation that maps n planes Π_i onto the matching planes Π_i' $(i = 1, \dots, n)$ amounts to finding the rotation matrix $\mathcal{R}$ that minimizes the error

$$E_r = \sum_{i=1}^{n} |\boldsymbol{n}_i' - \mathcal{R}\boldsymbol{n}_i|^2$$

and the translation vector $\boldsymbol{t}$ that minimizes

$$E_t = \sum_{i=1}^{n} (d_i' - d_i - \boldsymbol{n}_i' \cdot \boldsymbol{t})^2.$$

The rotation $\mathcal{R}$ minimizing E_r can be computed, exactly as in Section 24.4.1, by using the quaternion representation of matrices and solving an eigenvector problem.

The translation vector $\boldsymbol{t}$ minimizing E_t is the solution of a (non-homogeneous) linear least-squares problem, whose solution can be found using the techniques presented in Chapter 6.

Results

Figure 24.19 shows recognition results obtained using a bin of Renault parts such as the one shown in Figure 24.12. The range image of the bin has been segmented into planar patches using the technique presented in Section 24.2.2. The matching algorithm is run three times on the scene, with patches matched during each run removed from the scene before the next iteration. As shown by the figure, the three instances of the part present in the bin are correctly identified, and the accuracy of the pose estimation process is attested by the reprojection into the range image of the model in the computed pose.

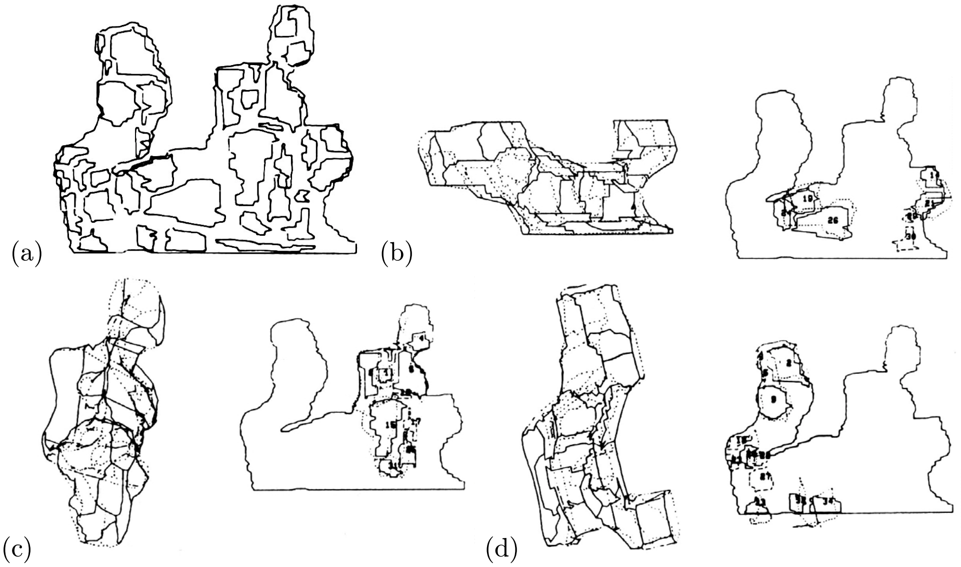

Figure 24.19. Recognition results: (a) a bin of parts, and (b)-(d) the three instances of the Renault part found in that bin. In each case, the model is shown both by itself in the position and orientation estimated by the algorithm, as well as superimposed (dotted lines) in this pose over the corresponding planes of the range image. Reprinted from [Faugeras and Hebert, 1986], Figures 14–16.

24.4.2 Matching Free-Form Surfaces Using Spin Images

As demonstrated in Section 24.2.1, differential geometry provides a powerful language for describing the shape of a surface *locally*, i.e., in a small neighborhood of each one of its points. On the other hand, the region-growing algorithm discussed in Section 24.2.2 is aimed at constructing a *globally* consistent surface description in terms of planar patches. We introduce in this section a *semi-local* surface representation, the spin image of Johnson and Hebert [1998; 1999], that captures the shape of a surface in a relatively large neighborhood of each one of its points. As will be shown in the rest of this section, the spin image is invariant under rigid transformations, and it affords an efficient algorithm for pointwise surface matching, thus completely bypassing segmentation in the recognition process.

Spin Image Definition

Let us assume as in Section 24.2.2 that the surface of interest is given in the form of a triangular mesh. The (outward-pointing) surface normal at each vertex can be estimated by fitting a plane to this vertex and its neighbors, turning the triangulation into a net of *oriented points*. Given an oriented point P, the *spin coordinates* of any other point Q can now be defined as the (nonnegative) distance α separating Q from the (oriented) normal line in P and the (signed) distance β from the tangent plane to Q (Figure 24.20).

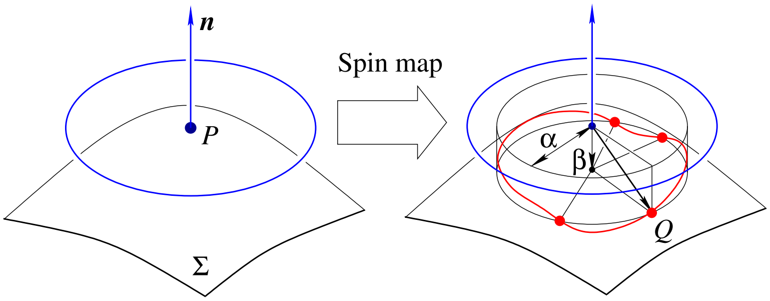

Figure 24.20. Definition of the spin map associated with a surface point P: the spin coordinates (α, β) of the point Q are respectively defined by the lengths of the projections of $\overrightarrow{PQ}$ onto the tangent plane and its surface normal. Note that there are three other points with the same (α, β) coordinates as Q in this example.

Accordingly, the *spin map* $s_P : \Sigma \to \mathbb{R}^2$ associated with P is defined for any point Q on Σ as

$$s_P(Q) \overset{\text{def}}{=} (\underbrace{|\overrightarrow{PQ} \times \boldsymbol{n}|}_{\alpha}, \underbrace{\overrightarrow{PQ} \cdot \boldsymbol{n}}_{\beta}).$$

As shown by Figure 24.20, this mapping is not injective. This is not surprising since the spin map only provides a partial specification of a cylindrical coordinate system: the third coordinate that would normally record the angle between some reference vector in the tangent plane and the projection of $\overrightarrow{PQ}$ into this plane is missing. The principal directions are obvious choices for such a reference vector, but focussing on the spin coordinates avoids their computation, a process that is susceptible to noise since it involves second derivatives and may be ambiguous for (almost) planar or spherical patches.

The *spin image* associated with an oriented point is a histogram of the α, β coordinates in a neighborhood of this point (Figure 24.20(b)). Concretely, the α, β plane is divided into a rectangular array of $\delta\alpha \times \delta\beta$ bins that accumulate the total surface area spanned by points with α, β values in that range.[3] As shown in [Carmichael *et al.*, 1999] and the exercises, each triangle in the surface mesh maps onto a region of the α, β plane whose boundaries are hyperbola arcs. Its contribution to the spin image can thus be computed by scan-converting this region and assigning to each bin that it traverses the area of the patch where the triangle intersects the annular region of $\mathbb{R}^3$ associated with the bin (Figure 24.21).

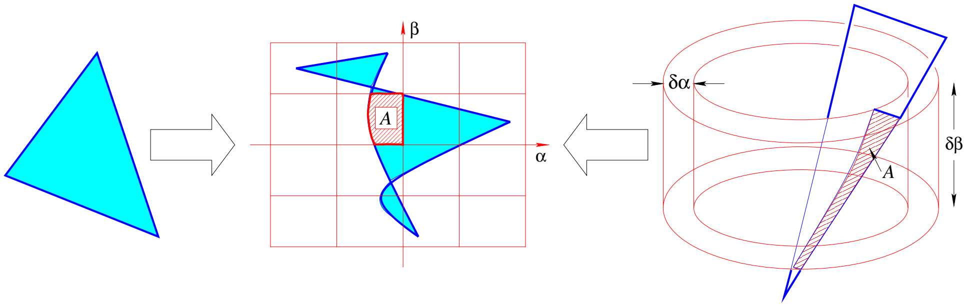

Figure 24.21. Spin image construction: the triangle shown in the left of the diagram maps onto a region with hyperbolic boundaries in the spin image; the value of each bin intersected by this region is incremented by the area of the portion of the triangle that intersects the annulus associated with the bin. After [Carmichael *et al.*, 1999, Figure 3].

A key parameter of spin images is the support distance d that limits to a sphere of radius d centered in P the range of the *support points* used to construct the image. This sphere must be large enough to provide good descriptive power but small enough to support recognition in the presence of clutter and occlusion. In practice, an appropriate choice for d might be a tenth of the object's diameter [Johnson and Hebert, 1998]: thus, as noted earlier, the spin image is indeed a semi-local description of the shape of a surface in an *extended* neighborhood of one of its points.

Robustness to clutter can be improved by limiting the range of surface normals at the support points to a cone of half-angle θ centered in $\boldsymbol{n}$. As in the support distance case, choosing the right value for θ involves a tradeoff between descriptive power and insensitivity to clutter; a value of 60° has empirically been shown to be satisfactory in [Johnson and Hebert, 1999]. The last parameter defining a spin

[3] The corresponding point sets may actually be divided into several connected components: for example, for small enough values of $\delta\alpha$ and $\delta\beta$ there are four connected components in the example shown in Figure 24.20, corresponding to small patches centered at the points having the same α, β coordinates as Q.

image is its size (in pixels), or equivalently, given the support distance, its bin size (in meters). As shown in [Johnson and Hebert, 1998], an appropriate choice for the bin size is the average distance between mesh vertices in the model. Figure 24.22 shows the spin images associated with three oriented points on the surface of a rubber duck.

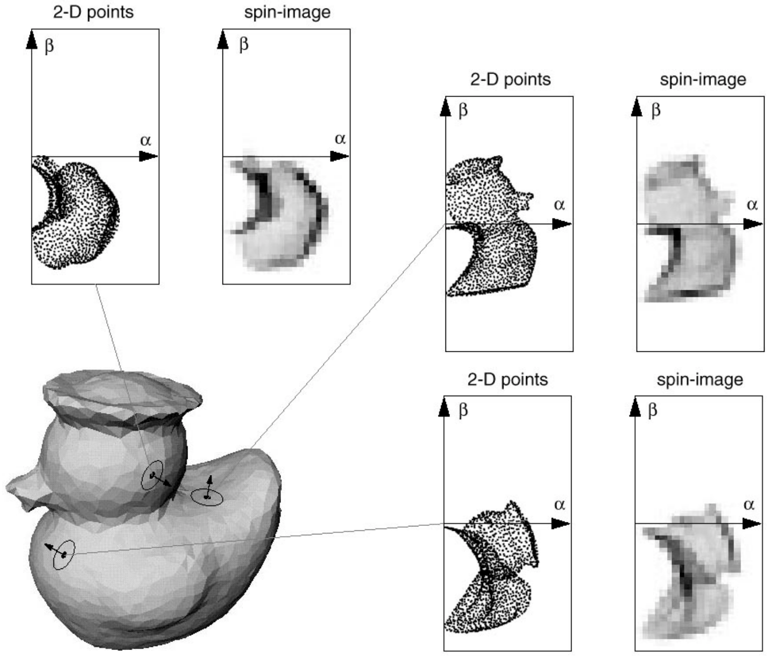

Figure 24.22. Three oriented points on the surface of a rubber duck and the corresponding spin images. The α, β coordinates of the mesh vertices are shown besides the actual spin images. Reprinted from [Johnson and Hebert, 1998], Figure 3.

Matching Spin Images

One of the most important features of spin images is that they are (obviously) invariant under rigid transformations. Thus an image comparison technique such as correlation can in principle be used to match the spin images associated with oriented points in the scene and the object model. Things are not that simple, however: we already noted that the spin map is not injective; in general, it is not surjective either, and empty bins (or equivalently zero-valued pixels) may occur for values of α and β that do not correspond to physical surface points (see the blank areas in Figure 24.22 for example). Occlusion may cause the appearance of zero pixels in the scene image, while clutter may introduce irrelevant non-empty bins. It is therefore reasonable to restrict the comparison of two spin images to their common non-zero pixels. In this context, Johnson and Hebert [1998] have shown that

$$S(\boldsymbol{I}, \boldsymbol{J}) \overset{\text{def}}{=} [\text{Arctanh}(C(\boldsymbol{I}, \boldsymbol{J}))]^2 - \frac{3}{N-3}$$

is an appropriate similarity measure for two spin images whose overlap regions contain N pixels and are represented by the vectors $\boldsymbol{I}$ and $\boldsymbol{J}$ of $\mathbb{R}^N$. In this formula, $C(\boldsymbol{I}, \boldsymbol{J})$ denotes the normalized correlation of the vectors $\boldsymbol{I}$ and $\boldsymbol{J}$, and Arctanh denotes the hyperbolic arc tangent function. Armed with this similarity measure, we can now outline a recognition algorithm that uses spin images to establish pointwise correspondences.

Off-line:

Compute the spin images associated with the oriented points of a surface model and store them into a table.

On-line:

1. Form correspondences between a set of spin images randomly selected in the scene and their best matches in the model table using the similarity measure S to rank-order the matches.

2. Filter and group correspondences using geometric consistency constraints, and compute the rigid transformations best aligning the matched scene and model features.

3. Verify the matches using the ICP algorithm.

Algorithm 24.4: *The algorithm of Johnson and Hebert [1998; 1999] for pointwise matching of free-form surfaces using spin images.*

The various stages of this algorithm are mostly straightforward. Let us note however that the filtering/grouping step relies on comparing the spin coordinates of model points relative to the other mesh vertices in their group with the spin coordinates of the corresponding scene points relative to their own group. Once consistent groups have been identified, an initial estimate of the rigid transformation aligning the scene and the model is computed from (oriented) point matches using the quaternion-based registration technique described in Section 24.3.1. Finally, consistent sets of correspondences are verified by iteratively spreading the matching process to their neighbors, updating along the way the rigid transformation that aligns the scene and the model.

Results

The matching algorithm presented in the previous section has been extensively tested in recognition tasks with cluttered indoor scenes that contain both industrial parts and various toys [Johnson and Hebert, 1998; Johnson and Hebert, 1999]. It has also been used in outdoor navigation/mapping tasks with very large datasets covering thousands of squared meters of terrain [Carmichael *et al.*, 1999]. Figure 24.23 shows sample recognition results in the toy domain.

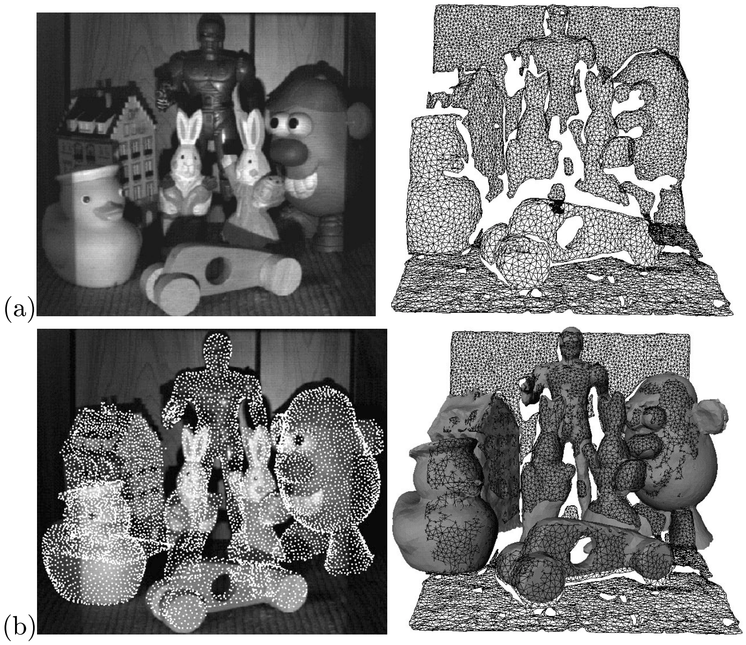

Figure 24.23. Spin-image recognition results: (a) a cluttered image of toys and the mesh constructed from the corresponding range image; (b) recognized objects overlaid on the original pictures.

24.5 Notes

Excellent surveys of active range finding techniques can be found in [Jarvis, 1983; Nitzan, 1988; Besl, 1989; Hebert, 2000]. The model-based approach to edge detection presented in Section 24.2.1 is only one of the many techniques that have been proposed for segmenting range pictures using notions from differential geometry (see, for example, [Fan *et al.*, 1987; Besl and Jain, 1988]). An alternative to the computational molecules used to smooth a range image in that section is provided by anisotropic diffusion, where the amount of smoothing at each point depends on the value of the gradient [Perona and Malik, 1990].

The method for segmenting surfaces into (almost) planar patches presented in Section 24.2.2 is easily extended to quadric patches (see [Faugeras and Hebert, 1986] and the exercises). Extensions to higher-order surface primitives is more problematic, in part because surface fitting is more difficult in that case. There is a vast amount of literature on the latter problem, using superquadrics (e.g., [Pentland, 1986; Bajcsy and Solina, 1987; Gross and Boult, 1988]) and algebraic surfaces (e.g, [Taubin *et al.*, 1994; Keren *et al.*, 1994; Sullivan *et al.*, 1994]) for example.

Alternatives to the Curless and Levoy [1996] approach to the fusion of multiple range images include the Delaunay triangulation algorithm of Boissonnat [1984], the zippered polygonal meshes of Turk and Levoy [1994] and the crust technique of Amenta *et al.* [1998]. The quaternion-based approach to the estimation of rigid transformations described in this chapter was developed independently by Faugeras and Hebert [1986] and Horn [1987]. The recognition technique discussed in Section 24.4.1 is closely related to other algorithms using interpretation trees to control the combinatorial cost of feature matching in the two- and three-dimensional cases [Gaston and Lozano-Pérez, 1984; Ayache and Faugeras, 1986; Grimson and Lozano-Pérez, 1987; Huttenlocher and Ullman, 1987].

The spin images discussed in Section 24.4.2 have been used to establish pointwise correspondences between range images and surface models. Related approaches to this problem include the structural indexing method of Stein and Medioni [1992] and the point signatures proposed by Chua and Jarvis [1996]. A (local) variant of the same idea will be discussed in Chapter 21 in the context of object recognition from photographs [Schmid and Mohr, 1997a]. To conclude, let us note that the original algorithm described in Section 24.4.2 has been extended in various directions: a scene can now be matched simultaneously to several models using principal component analysis (see Chapter 21 and [Johnson and Hebert, 1999]), while learning techniques are used to prune false matches in cluttered scenes [Carmichael *et al.*, 1999].

24.6 Assignments

Exercises

1. Step model: compute $z_\sigma(x) = G_\sigma * z(x)$, where $z(x)$ is given by (24.2.2). Show that z''_σ is given by (24.2.3). Conclude that $\kappa''_\sigma/\kappa'_\sigma = -2\delta/h$ in the point x_σ where z''_σ and κ_σ vanish.

2. Roof model: show that κ_σ is given by (24.2.4).

3. Use (24.2.1) to show that a necessary and sufficient condition for the coordinate curves of a parameterized surface to be principal directions is that $f = F = 0$.

4. Show that the lines of curvature of a surface of revolution are its meridians and parallels.

5. Calculate the Gaussian curvature of an SHGC.

6. Show that the matrix $\mathcal{A}_i$ constructed in Section 24.3.1 is equal to

$$\mathcal{A}_i = \begin{pmatrix} 0 & \boldsymbol{y}_i^T - \boldsymbol{y}_i'^T \\ \boldsymbol{y}_i' - \boldsymbol{y}_i & [\boldsymbol{y}_i + \boldsymbol{y}_i']_\times \end{pmatrix}.$$

7. As mentioned earlier, the ICP method can be extended to various types of geometric models. We consider here the case of polyhedral models and piecewise parametric patches.

   (a) Give a method for computing the point Q in a polygon that is closest to some point P.

   (b) Give a method for computing the point Q in the parametric patch $\boldsymbol{x} : I \times J \to \mathbb{R}^3$ that is closest to some point P. Hint: use Newton iterations.

8. Develop a linear least-squares method for fitting a quadric surface to a set of points under the constraint that the quadratic form has unit Frobenius form.

9. Show that a surface triangle maps onto a patch with hyperbolic edges in α, β space.

Programming Assignments

The datasets for the following machine problems can be found on the CD companion of this book.

1. Implement molecule-based smoothing and the computation of principal directions and curvatures.

2. Implement the region-growing approach to plane segmentation described in this chapter.

3. Implement an algorithm for computing the lines of curvature of a surface from its range image. Hint: use a curve-growing algorithm analogous to the region-growing algorithm for plane segmentation.

4. Implement the Besl-McKay ICP registration algorithm.

5. Marching squares in the plane: develop and implement an algorithm for finding the zero set of a planar density function. Hint: work out the possible ways a curve may intersect the edges of a pixel, and use linear interpolation along these edges to identify the zero set.

6. Implement the registration part of the Faugeras-Hebert algorithm.