

## 6 3-D Modeling of Real-World Objects Using Range and Intensity Images

Johnny Park<sup>1</sup>, Guilherme N. DeSouza<sup>2</sup>

1. School of Electrical and Computer Engineering, Purdue University, West Lafayette, Indiana, U.S.A.  
jpark@purdue.edu
2. School of Electrical, Electronic & Computer Engineering, The University of Western Australia, Australia  
gdesouza@ee.uwa.edu.au

### 6.1 Introduction

In the last few decades, constructing accurate three-dimensional models of real-world objects has drawn much attention from many industrial and research groups. Earlier, the 3D models were used primarily in robotics and computer vision applications such as bin picking and object recognition. The models for such applications only require salient geometric features of the objects so that the objects can be recognized and the pose determined. Therefore, it is unnecessary in these applications for the models to faithfully capture every detail on the object surface. More recently, however, there has been considerable interest in the construction of 3D models for applications where the focus is more on visualization of the object by humans. This interest is fueled by the recent technological advances in range sensors, and the rapid increase of computing power that now enables a computer to represent an object surface by millions of polygons which allows such representations to be visualized interactively in real-time. Obviously, to take advantage of these technological advances, the 3D models constructed must capture to the maximum extent possible of the shape and surface-texture information of real-world objects. By real-world objects, we mean objects that may present self-occlusion with respect to the sensory devices; objects with shiny surfaces that may create mirror-like (specular) effects; objects that may absorb light and therefore not be completely perceived by the vision system; and other types of optically uncooperative objects. Construction of such photo-realistic 3D models of real-world objects is the main focus of this chapter. In general, the construction of such 3D models entails four main steps:

#### 1. Acquisition of geometric data:

First, a range sensor must be used to acquire the geometric shape of the exterior of the object. Objects of complex shape may require a large number of range images viewed from different directions so that all of

the surface detail is captured, although it is very difficult to capture the entire surface if the object contains significant protrusions.

2. **Registration:**

The second step in the construction is the registration of the multiple range images. Since each view of the object that is acquired is recorded in its own coordinate frame, we must register the multiple range images into a common coordinate system called the world frame.

3. **Integration:**

The registered range images taken from adjacent viewpoints will typically contain overlapping surfaces with common features in the areas of overlap. This third step consists of integrating the registered range images into a single connected surface model; this process first takes advantage of the overlapping portions to determine how the different range images fit together and then eliminates the redundancies in the overlap areas.

4. **Acquisition of reflection data:**

In order to provide a photo-realistic visualization, the final step acquires the reflectance properties of the object surface, and this information is added to the geometric model.

Each of these steps will be described in separate sections of this chapter.

## 6.2 Acquisition of Geometric Data

The first step in 3D object modeling is to acquire the geometric shape of the exterior of the object. Since acquiring geometric data of an object is a very common problem in computer vision, various techniques have been developed over the years for different applications.

### 6.2.1 Techniques of Acquiring 3D Data

The techniques described in this section are not intended to be exhaustive; we will mention briefly only the prominent approaches. In general, methods of acquiring 3D data can be divided into passive sensing methods and active sensing methods.

#### *Passive Sensing Methods*

The passive sensing methods extract 3D positions of object points by using images with ambient light source. Two of the well-known passive sens-

ing methods are Shape-From-Shading (SFS) and stereo vision. The Shape-From-Shading method uses a single image of an object. The main idea of this method derives from the fact that one of the cues the human visual system uses to infer the shape of a 3D object is its shading information. Using the variation in brightness of an object, the SFS method recovers the 3D shape of an object. There are three major drawbacks of this method: First, the shadow areas of an object cannot be recovered reliably since they do not provide enough intensity information. Second, the method assumes that the entire surface of an object has uniform reflectance property, thus the method cannot be applied to general objects. Third, the method is very sensitive to noise since the computation of surface gradients is involved.

The stereo vision method uses two or more images of an object from different viewpoints. Given the image coordinates of the same object point in two or more images, the stereo vision method extracts the 3D coordinate of that object point. A fundamental limitation of this method is the fact that finding the correspondence between images is extremely difficult.

The passive sensing methods require very simple hardware, but usually these methods do not generate dense and accurate 3D data compare to the active sensing methods.

### ***Active Sensing Methods***

The active sensing methods can be divided into two categories: contact and non-contact methods. Coordinate Measuring Machine (CMM) is a prime example of the contact methods. CMMs consist of probe sensors which provide 3D measurements by touching the surface of an object. Although CMMs generate very accurate and fine measurements, they are very expensive and slow. Also, the types of objects that can be used by CMMs are limited since physical contact is required.

The non-contact methods project their own energy source to an object, then observe either the transmitted or the reflected energy. The computed tomography (CT), also known as the computed axial tomography (CAT), is one of the techniques that records the transmitted energy. It uses X-ray beams at various angles to create cross-sectional images of an object. Since the computed tomography provides the internal structure of an object, the method is widely used in medical applications.

The active stereo uses the same idea of the passive sensing stereo method, but a light pattern is projected onto an object to solve the difficulty of finding corresponding points between two (or more) camera images.

The laser radar system, also known as LADAR, LIDAR, or optical radar, uses the information of emitted and received laser beam to compute the depth. There are mainly two methods that are widely used: (1) using amplitude modulated continuous wave (AM-CW) laser, and (2) using laser pulses.

The first method emits AM-CW laser onto a scene, and receives the laser that was reflected by a point in the scene. The system computes the phase difference between the emitted and the received laser beam. Then, the depth of the point can be computed since the phase difference is directly proportional to depth. The second method emits a laser pulse, and computes the interval between the emitted and the received time of the pulse. The time interval, well known as *time-of-flight*, is then used to compute the depth given by  $t = 2z/c$  where  $t$  is time-of-flight,  $z$  is depth, and  $c$  is speed of light. The laser radar systems are well suited for applications requiring medium-range sensing from 10 to 200 meters.

The structured-light methods project a light pattern onto a scene, then use a camera to observe how the pattern is illuminated on the object surface. Broadly speaking, the structured-light methods can be divided into scanning and non-scanning methods. The scanning methods consist of a moving stage and a laser plane, so either the laser plane scans the object or the object moves through the laser plane. A sequence of images is taken while scanning. Then, by detecting illuminated points in the images, 3D positions of corresponding object points are computed by the equations of camera calibration. The non-scanning methods project a spatially or temporally varying light pattern onto an object. An appropriate decoding of the reflected pattern is then used to compute the 3D coordinates of an object.

The system that acquired all the 3D data presented in this chapter falls into a category of a scanning structured-light method using a single laser plane. From now on, such a system will be referred to as a structured-light scanner.

### 6.2.2 Structured-Light Scanner

Structured-light scanners have been used in manifold applications since the technique was introduced about two decades ago. They are especially suitable for applications in 3D object modeling for two main reasons: First, they acquire dense and accurate 3D data compared to passive sensing methods. Second, they require relatively simple hardware compared to laser radar systems.

In what follows, we will describe the basic concept of structured-light scanner and all the data that can be typically acquired and derived from this kind of sensor.

#### ***A Typical System***

A sketch of a typical structured-light scanner is shown in Figure 6.1. The system consists of four main parts: linear stage, rotary stage, laser projector, and camera. The linear stage moves along the  $X$  axis and the rotary stage mounted on top of the linear stage rotates about the  $Z$  axis where  $XYZ$  are

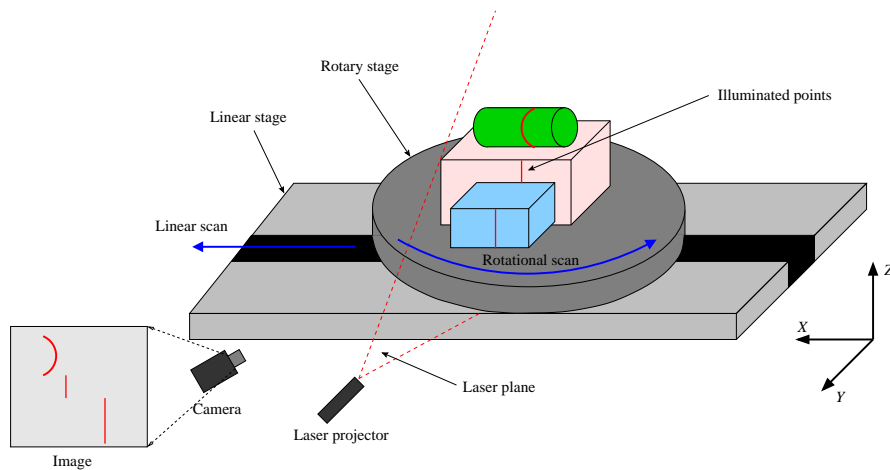


Fig. 6.1: A typical structured-light scanner.

the three principle axes of the reference coordinate system. A laser plane parallel to the  $YZ$  plane is projected onto the objects. The intersection of the laser plane and the objects creates a stripe of illuminated points on the surface of the objects. The camera captures the scene, and the illuminated points in that image are extracted. Given the image coordinates of the extracted illuminated points and the positions of the linear and rotary stages, the corresponding 3D coordinates with respect to the reference coordinate system can be computed by the equations of camera calibration; we will describe the process of camera calibration shortly. Such process only acquires a set of 3D coordinates of the points that are illuminated by the laser plane. In order to capture the entire scene, the system either translates or rotates the objects through the laser plane while the camera takes the sequence of images. Note that it is possible to have the objects stationary, and move the sensors (laser projector and camera) to sweep the entire scene.

#### **Acquiring Data: Range Image**

The sequence of images taken by the camera during a scan can be stored in a more compact data structure called *range image*, also known as *range map*, *range data*, *depth map*, or *depth image*. A range image is a set of distance measurements arranged in a  $m \times n$  grid. Typically, for the case of structured-light scanner,  $m$  is the number of horizontal scan lines (rows) of camera image, and  $n$  is the total number of images (i.e., number of stripes) in the sequence. We can also represent a range image in a parametric form  $r(i, j)$  where  $r$  is the column coordinate of the illuminated point at the  $i$ th

row in the  $j$ th image. Sometimes, the computed 3D coordinate  $(x, y, z)$  is stored instead of the column coordinate of the illuminated point. Typically, the column coordinates of the illuminated points are computed in a sub-pixel accuracy as will be described next. If an illuminated point cannot be detected, a special number (e.g., -1) can be assigned to the corresponding entry indicating that no data is available. An example of a range image is depicted in Figure 6.2.

Assuming a range image  $r(i, j)$  is acquired by the system shown in Figure 6.1,  $i$  is related mainly to the coordinates along the  $Z$  axis of the reference coordinate system,  $j$  the  $X$  axis, and  $r$  the  $Y$  axis. Since a range image is maintained in a grid, the neighborhood information is directly provided. That is, we can easily obtain the closest neighbors for each point, and even detect spatial discontinuity of the object surface. This is very useful especially for computing normal directions of each data point, or generating triangular mesh; the discussion of these topics will follow shortly.

### **Computing Center of Illuminated Points**

In order to create the range images as described above, we must collect one (the center) of the illuminated points in each row as the representative of that row. Assuming the calibrations of both the camera and the positioning stages are perfect, the accuracy of computing 3D coordinates of object points primarily depends on locating the true center of these illuminated points. A typical intensity distribution around the illuminated points is shown in Figure 6.3.

Ideally only the light source (e.g., laser plane) should cause the illumination, and the intensity curve around the illuminated points should be Gaussian. However, we need to be aware that the illumination may be affected by many different factors such as: CCD camera error (e.g., noise and quantization error); laser speckle; blurring effect of laser; mutual-reflections of object surface; varying reflectance properties of object surface; high curvature on object surface; partial occlusions with respect to camera or laser plane; etc. Although eliminating all these sources of error is unlikely, it is important to use an algorithm that will best estimate the true center of the illuminated points.

Here we introduce three algorithms: (1) center of mass, (2) Blais and Rioux algorithm, and (3) Gaussian approximation. Let  $I(i)$  be the intensity value at  $i$  coordinate, and let  $p$  be the coordinate with peak intensity. Then, each algorithm computes the center  $c$  as follows:

1. **Center of mass:** This algorithm solves the location of the center by computing weighted average. The size of kernel  $n$  should be set such

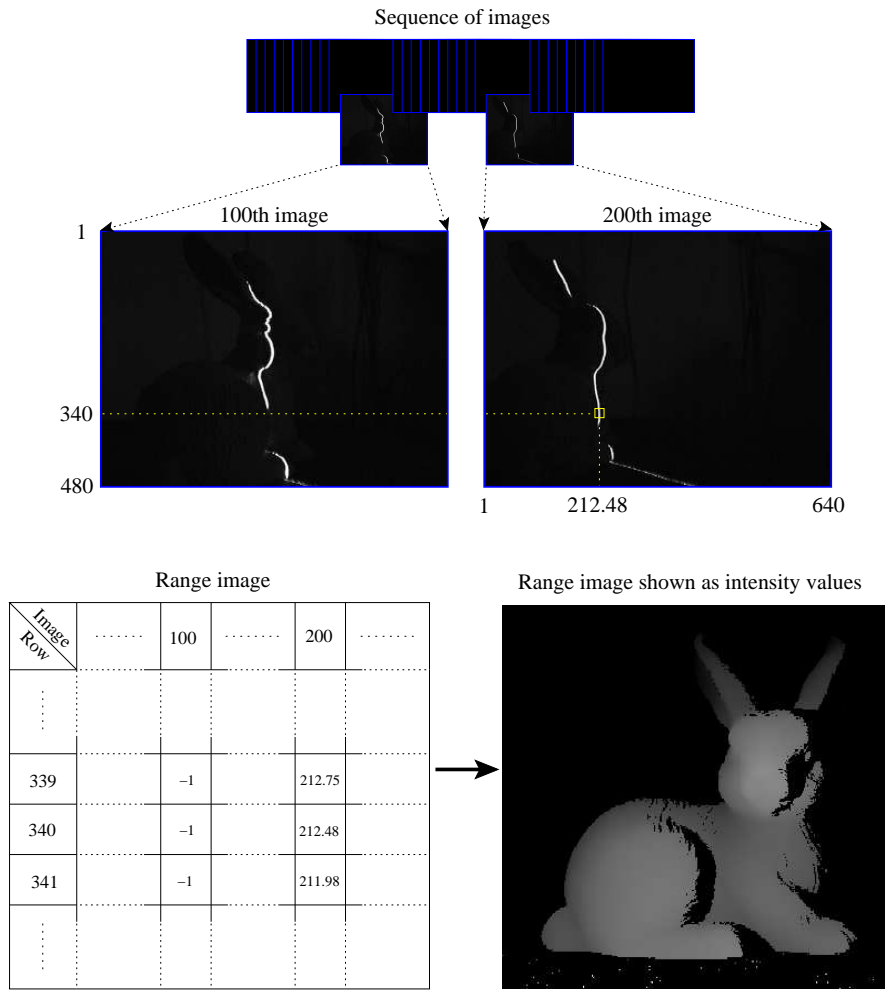


Fig. 6.2: Converting a sequence of images into a range image

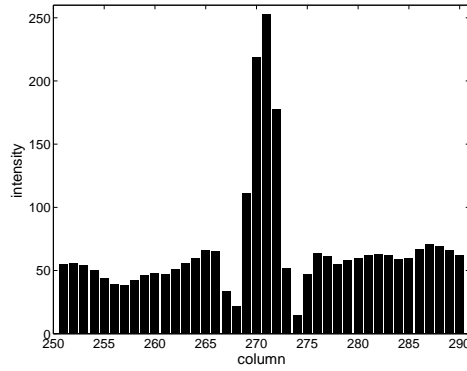


Fig. 6.3: Typical intensity distribution around illuminated points.

that all illuminated points are included.

$$c = \frac{\sum_{i=p-n}^{p+n} iI(i)}{\sum_{i=p-n}^{p+n} I(i)}$$

2. **Blais and Rioux algorithm** [9]: This algorithm uses a finite impulse response filter to differentiate the signal and to eliminate the high frequency noise. The zero crossing of the derivative is linearly interpolated to solve the location of the center.

$$c = p + \frac{h(p)}{h(p) - h(p+1)}$$

where  $h(i) = I(i-2) + I(i-1) - I(i+1) - I(i+2)$ .

3. **Gaussian approximation** [55]: This algorithm fits a Gaussian profile to three contiguous intensities around the peak.

$$c = p - \frac{1}{2} \frac{\ln(I(p+1)) - \ln(I(p-1))}{\ln(I(p-1)) - 2\ln(I(p)) + \ln(I(p+1))}$$

After testing all three methods, one would notice that the center of mass method produces the most reliable results for different objects with varying reflection properties. Thus, all experimental results shown in this chapter were obtained using the center of mass method.

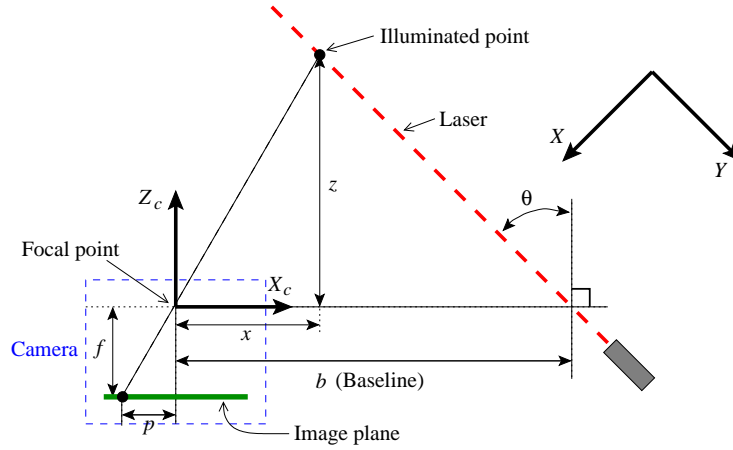


Fig. 6.4: Optical triangulation

### Optical Triangulation

Once the range image is complete, we must now calculate the 3D structure of the scanned object. The measurement of the depth of an object using a structured-light scanner is based on optical triangulation. The basic principles of optical triangulation are depicted in Figure 6.4.  $X_c$  and  $Z_c$  are two of the three principle axes of the camera coordinate system,  $f$  is the focal length of the camera,  $p$  is the image coordinate of the illuminated point, and  $b$  (baseline) is the distance between the focal point and the laser along the  $X_c$  axis. Notice that the figure corresponds to the top view of the structured-light scanner in Figure 6.1.

Using the notations in Figure 6.4, the following equation can be obtained by the properties of similar triangles:

$$\frac{z}{f} = \frac{b}{p + f \tan \theta} \quad (1)$$

Then, the  $z$  coordinate of the illuminated point with respect to the camera coordinate system is directly given by

$$z = \frac{fb}{p + f \tan \theta} \quad (2)$$

Given the  $z$  coordinate, the  $x$  coordinate can be computed as

$$x = b - z \tan \theta \quad (3)$$

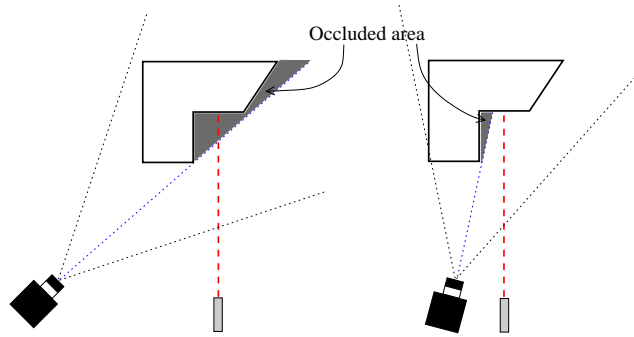


Fig. 6.5: Tradeoff between the length of baseline and the occlusion. As the length of baseline increases, a better accuracy in the measurement can be achieved, but the occluded area due to shadow effect becomes larger, and vice versa.

The error of  $z$  measurement can be obtained by differentiating Eq. (2):

$$\Delta z = \frac{fb}{(p + f \tan \theta)^2} \Delta p + \frac{fb (f \sec^2 \theta)}{(p + f \tan \theta)^2} \Delta \theta \quad (4)$$

where  $\Delta p$  and  $\Delta \theta$  are the measurement errors of  $p$  and  $\theta$  respectively. Substituting the square of Eq. (2), we now have

$$\Delta z = \frac{z^2}{fb} \Delta p + \frac{z^2 \sec^2 \theta}{b} \Delta \theta \quad (5)$$

This equation indicates that the error of the  $z$  measurement is directly proportional to the square of  $z$ , but inversely proportional to the focal length  $f$  and the baseline  $b$ . Therefore, increasing the baseline implies a better accuracy in the measurement. Unfortunately, the length of baseline is limited by the hardware structure of the system, and there is a tradeoff between the length of baseline and the sensor occlusions – as the length of baseline increases, a better accuracy in the measurement can be achieved, but the occluded area due to shadow effect becomes larger, and vice versa. A pictorial illustration of this tradeoff is shown in Figure 6.5.

### Computing 3D World Coordinates

The coordinates of illuminated points computed by the equations of optical triangulation are with respect to the camera coordinate system. Thus, an additional transformation matrix containing the extrinsic parameters of the camera (i.e., a rotation matrix and a translation vector) that transforms the camera coordinate system to the reference coordinate system needs to be

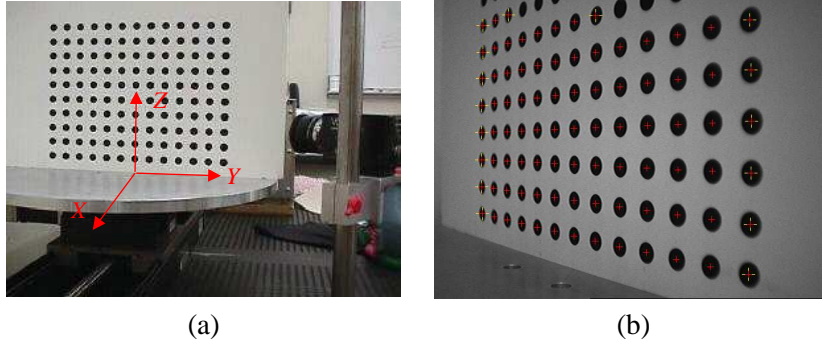


Fig. 6.6: Calibration pattern

(a): A calibration pattern is placed in such a way that the pattern surface is parallel to the laser plane (i.e.,  $YZ$  plane), and the middle column of the pattern (i.e., 7th column) coincides the  $Z$  axis of the reference coordinate system. (b): Image taken from the camera. Crosses indicate extracted centers of circle patterns.

found. However, one can formulate a single transformation matrix that contains the optical triangulation parameters and the camera calibration parameters all together. In fact, the main reason we derived the optical triangulation equations is to show that the uncertainty of depth measurement is related to the square of the depth, focal length of the camera, and the baseline.

The transformation matrix for computing 3D coordinates with respect to the reference coordinate system can be obtained as follows. Suppose we have  $n$  data points with known reference coordinates and the corresponding image coordinates. Such points can be obtained by using a calibration pattern placed in a known location, for example, the pattern surface is parallel to the laser plane and the middle column of the pattern coincides the  $Z$  axis (See Figure 6.6).

Let the reference coordinate of the  $i$ th data point be denoted by  $(x_i, y_i, z_i)$ , and the corresponding image coordinate be denoted by  $(u_i, v_i)$ . We want to solve a matrix  $\mathbf{T}$  that transforms the image coordinates to the reference coordinates. It is well known that the homogeneous coordinate system must be used for linearization of 2D to 3D transformation. Thus, we can formulate the transformation as

$$\mathbf{T}_{4 \times 3} \begin{bmatrix} u_i \\ v_i \\ 1 \end{bmatrix} = \begin{bmatrix} \bar{x}_i \\ \bar{y}_i \\ \bar{z}_i \\ \rho \end{bmatrix} \quad (6)$$

or

$$\begin{bmatrix} t_{11} & t_{12} & t_{13} \\ t_{21} & t_{22} & t_{23} \\ t_{31} & t_{32} & t_{33} \\ t_{41} & t_{42} & t_{43} \end{bmatrix} \begin{bmatrix} u_i \\ v_i \\ 1 \end{bmatrix} = \begin{bmatrix} \overline{x_i} \\ \overline{y_i} \\ \overline{z_i} \\ \rho \end{bmatrix} \quad (7)$$

where

$$\begin{bmatrix} x_i \\ y_i \\ z_i \end{bmatrix} = \begin{bmatrix} \overline{x_i} / \rho \\ \overline{y_i} / \rho \\ \overline{z_i} / \rho \end{bmatrix} \quad (8)$$

We use the free variable  $\rho$  to account for the non-uniqueness of the homogeneous coordinate expressions (i.e., scale factor). Carrying out the first row and the fourth row of Eq. (7), we have

$$\begin{aligned} \overline{x_1} &= t_{11}u_1 + t_{12}v_1 + t_{13} \\ \overline{x_2} &= t_{11}u_2 + t_{12}v_2 + t_{13} \\ &\vdots \quad \quad \quad \vdots \\ \overline{x_n} &= t_{11}u_n + t_{12}v_n + t_{13} \end{aligned} \quad (9)$$

and

$$\begin{aligned} \rho &= t_{41}u_1 + t_{42}v_1 + t_{43} \\ \rho &= t_{41}u_2 + t_{42}v_2 + t_{43} \\ &\vdots \quad \quad \quad \vdots \\ \rho &= t_{41}u_n + t_{42}v_n + t_{43} \end{aligned} \quad (10)$$

By combining these two sets of equations, and by setting  $\overline{x_i} - \rho x_i = 0$ , we obtain

$$\begin{aligned} t_{11}u_1 + t_{12}v_1 + t_{13} - t_{41}u_1x_1 - t_{42}v_1x_1 - t_{43}x_1 &= 0 \\ t_{11}u_2 + t_{12}v_2 + t_{13} - t_{41}u_2x_2 - t_{42}v_2x_2 - t_{43}x_2 &= 0 \\ &\vdots \quad \quad \quad \vdots \\ t_{11}u_n + t_{12}v_n + t_{13} - t_{41}u_nx_n - t_{42}v_nx_n - t_{43}x_n &= 0 \end{aligned} \quad (11)$$

Since we have a free variable  $\rho$ , we can set  $t_{43} = 1$  which will appropriately scale the rest of the variables in the matrix  $\mathbf{M}$ . Carrying out the same procedure that produced Eq. (11) for  $y_i$  and  $z_i$ , and rearranging all the equations into a matrix form, we obtain

$$\begin{bmatrix}
 u_1 & v_1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & -u_1x_1 & -v_1x_1 \\
 u_2 & v_2 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & -u_2x_2 & -v_2x_2 \\
 \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\
 u_n & v_n & 1 & 0 & 0 & 0 & 0 & 0 & 0 & -u_nx_n & -v_nx_n \\
 0 & 0 & 0 & u_1 & v_1 & 1 & 0 & 0 & 0 & -u_1y_1 & -v_1y_1 \\
 0 & 0 & 0 & u_2 & v_2 & 1 & 0 & 0 & 0 & -u_2y_2 & -v_2y_2 \\
 \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\
 0 & 0 & 0 & u_n & v_n & 1 & 0 & 0 & 0 & -u_ny_n & -v_ny_n \\
 0 & 0 & 0 & 0 & 0 & 0 & u_1 & v_1 & 1 & -u_1z_1 & -v_1z_1 \\
 0 & 0 & 0 & 0 & 0 & 0 & u_2 & v_2 & 1 & -u_2z_2 & -v_2z_2 \\
 \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\
 0 & 0 & 0 & 0 & 0 & 0 & u_n & v_n & 1 & -u_nz_n & -v_nz_n
 \end{bmatrix}
 \begin{bmatrix}
 t_{11} \\
 t_{12} \\
 t_{13} \\
 t_{21} \\
 t_{22} \\
 t_{23} \\
 t_{31} \\
 t_{32} \\
 t_{33} \\
 t_{41} \\
 t_{42}
 \end{bmatrix}
 =
 \begin{bmatrix}
 x_1 \\
 x_2 \\
 \vdots \\
 x_n \\
 y_1 \\
 y_2 \\
 \vdots \\
 y_n \\
 z_1 \\
 z_2 \\
 \vdots \\
 z_n
 \end{bmatrix}
 \quad (12)$$

If we rewrite Eq. (12) as  $\mathbf{Ax} = \mathbf{b}$ , then our problem is to solve for  $\mathbf{x}$ . We can form the normal equations and find the linear least squares solution by solving  $(\mathbf{A}^T \mathbf{A})\mathbf{x} = \mathbf{A}^T \mathbf{b}$ . The resulting solution  $\mathbf{x}$  forms the transformation matrix  $\mathbf{T}$ . Note that Eq. (12) contains  $3n$  equations and 11 unknowns, therefore the minimum number of data points needed to solve this equation is 4.

Given the matrix  $\mathbf{T}$ , we can now compute 3D coordinates for each entry of a range image. Let  $\mathbf{p}(i, j)$  represent the 3D coordinates  $(x, y, z)$  of a range image entry  $r(i, j)$  with respect to the reference coordinate system; recall that  $r(i, j)$  is the column coordinate of the illuminated point at the  $i$ th row in the  $j$ th image. Using Eq. (6), we have

$$\begin{bmatrix}
 \bar{x} \\
 \bar{y} \\
 \bar{z} \\
 \rho
 \end{bmatrix}
 =
 \mathbf{T}
 \begin{bmatrix}
 i \\
 r(i, j) \\
 1
 \end{bmatrix}
 \quad (13)$$

and the corresponding 3D coordinate is computed by

$$\mathbf{p}(i, j) =
 \begin{bmatrix}
 \bar{x} / \rho \\
 \bar{y} / \rho \\
 \bar{z} / \rho
 \end{bmatrix}
 +
 \begin{bmatrix}
 x_0 + (j - 1)\Delta x \\
 0 \\
 0
 \end{bmatrix}
 \quad (14)$$

where  $x_0$  is the  $x$  coordinate of the laser plane at the beginning of the scan, and  $\Delta x$  is the distance that the linear slide moved along the  $X$  axis between two consecutive images.

The transformation matrix  $\mathbf{T}$  computed by Eq. (12) is based on the assumption that the camera image plane is perfectly planar, and that all the data points are linearly projected onto the image plane through an infinitely small focal point. This assumption, often called as *pin-hole camera model*,

generally works well when using cameras with normal lenses and small calibration error is acceptable. However, when using cameras with wide-angle lenses or large aperture, and a very accurate calibration is required, this assumption may not be appropriate. In order to improve the accuracy of camera calibration, two types of camera lens distortions are commonly accounted for: radial distortion and decentering distortion. Radial distortion is due to flawed radial curvature curve of the lens elements, and it causes inward or outward perturbations of image points. Decentering distortion is caused by non-collinearity of the optical centers of lens elements. The effect of the radial distortion is generally much more severe than that of the decentering distortion.

In order to account for the lens distortions, a simple transformation matrix can no longer be used; we need to find both the intrinsic and extrinsic parameters of the camera as well as the distortion parameters. A widely accepted calibration method is Tsai's method, and we refer the readers to [56, 34] for the description of the method.

### **Computing Normal Vectors**

Surface normal vectors are important to the determination of the shape of an object, therefore it is necessary to estimate them reliably. Given the 3D coordinate  $\mathbf{p}(i, j)$  of the range image entry  $r(i, j)$ , its normal vector  $\mathbf{n}(i, j)$  can be computed by

$$\mathbf{n}(i, j) = \frac{\frac{\partial \mathbf{p}}{\partial i} \times \frac{\partial \mathbf{p}}{\partial j}}{\left\| \frac{\partial \mathbf{p}}{\partial i} \times \frac{\partial \mathbf{p}}{\partial j} \right\|} \quad (15)$$

where  $\times$  is a cross product. The partial derivatives can be computed by finite difference operators. This approach, however, is very sensitive to noise due to the differentiation operations. Some researchers have tried to overcome the noise problem by smoothing the data, but it causes distortions to the data especially near sharp edges or high curvature regions.

An alternative approach computes the normal direction of the plane that best fits some neighbors of the point in question. In general, a small window (e.g.,  $3 \times 3$ , or  $5 \times 5$ ) centered at the point is used to obtain the neighboring points, and the PCA (Principal Component Analysis) for computing the normal of the best fitting plane.

Suppose we want to compute the normal vector  $\mathbf{n}(i, j)$  of the point  $\mathbf{p}(i, j)$  using a  $n \times n$  window. The center of mass  $\mathbf{m}$  of the neighboring points is computed by

$$\mathbf{m} = \frac{1}{n^2} \sum_{r=i-a}^{i+a} \sum_{c=j-a}^{j+a} \mathbf{p}(r, c) \quad (16)$$

where  $a = \lfloor n/2 \rfloor$ . Then, the covariance matrix  $\mathbf{C}$  is computed by

$$\mathbf{C} = \sum_{r=i-a}^{i+a} \sum_{c=j-a}^{j+a} [\mathbf{p}(r, c) - \mathbf{m}] [\mathbf{p}(r, c) - \mathbf{m}]^T \quad (17)$$

The surface normal is estimated as the eigenvector with the smallest eigenvalue of the matrix  $\mathbf{C}$ .

Although using a fixed sized window provides a simple way of finding neighboring points, it may also cause the estimation of normal vectors to become unreliable. This is the case when the surface within the fixed window contains noise, a crease edge, a jump edge, or simply missing data. Also, when the vertical and horizontal sampling resolutions of the range image are significantly different, the estimated normal vectors will be less robust with respect to the direction along which the sampling resolution is lower. Therefore, a region growing approach can be used for finding the neighboring points. That is, for each point of interest, a continuous region is defined such that the distance between the point of interest to each point in the region is less than a given threshold. Taking the points in the region as neighboring points reduces the difficulties mentioned above, but obviously requires more computations. The threshold for the region growing can be set, for example, as  $2(v+h)$  where  $v$  and  $h$  are the vertical and horizontal sampling resolutions respectively.

### **Generating Triangular Mesh from Range Image**

Generating triangular mesh from a range image is quite simple since a range image is maintained in a regular grid. Each sample point (entry) of a  $m \times n$  range image is a potential vertex of a triangle. Four neighboring sample points are considered at a time, and two diagonal distances  $d_{14}$  and  $d_{23}$  as in Figure 6.7(a) are computed. If both distances are greater than a threshold, then no triangles are generated, and the next four points are considered. If one of the two distances is less than the threshold, say  $d_{14}$ , we have potentially two triangles connecting the points 1-3-4 and 1-2-4. A triangle is created when the distances of all three edges are below the threshold. Therefore, either zero, one, or two triangles are created with four neighboring points. When both diagonal distances are less than the threshold, the diagonal edge with the smaller distance is chosen. Figure 6.7(b) shows an example of the triangular mesh using this method.

The distance threshold is, in general, set to a small multiple of the sam-

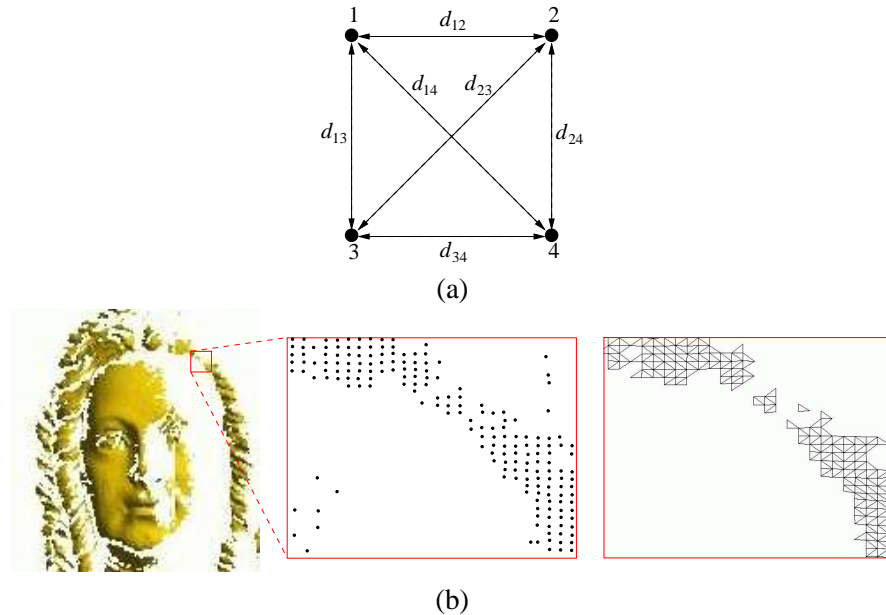


Fig. 6.7: Triangulation of range image

pling resolution. As illustrated in Figure 6.8, triangulation errors are likely to occur on object surfaces with high curvature, or on surfaces where the normal direction is close to the perpendicular to the viewing direction from the sensor. In practice, the threshold must be small enough to reject false edges even if it means that some of the edges that represent true surfaces can also be rejected. That is because we can always acquire another range image from a different viewing direction that can sample those missing surfaces more densely and accurately; however, it is not easy to remove false edges once they are created.

### **Experimental Result**

To illustrate all the steps described above, we present the result images obtained in our lab. Figure 6.9 shows a photograph of our structured-light scanner. The camera is a Sony XC-7500 with pixel resolution of 659 by 494. The laser has  $685\text{nm}$  wavelength with  $50\text{mW}$  diode power. The rotary stage is Aerotech ART310, the linear stage is Aerotech ATS0260 with  $1.25\mu\text{m}$  resolution and  $1.0\mu\text{m}/25\text{mm}$  accuracy, and these stages are controlled by Aerotech Unidex 511.

Figure 6.10 shows the geometric data from a single linear scan acquired

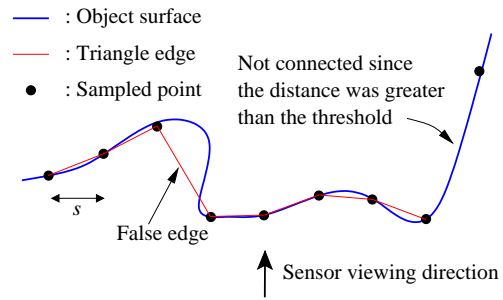


Fig. 6.8: Problems with triangulation

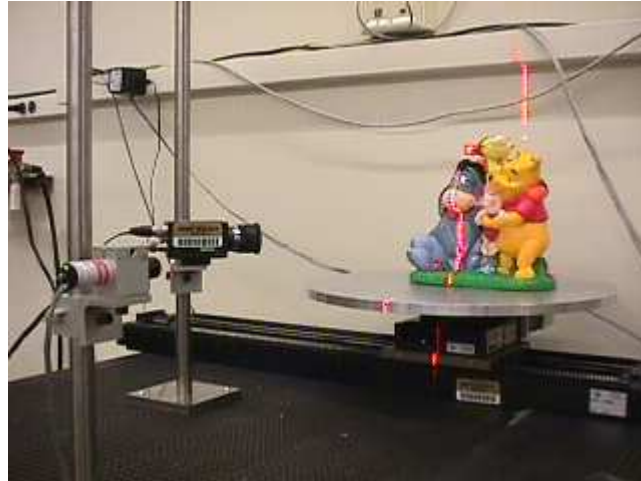


Fig. 6.9: Photograph of our structured-light scanner

by our structured-light scanner. Figure 6.10(a) shows the photograph of the object that was scanned, (b) shows the range image displayed as intensity values, (c) shows the computed 3D coordinates as point cloud, (d) shows the shaded triangular mesh, and finally (e) shows the normal vectors displayed as *RGB* colors where the *X* component of the normal vector corresponds to the *R* component, the *Y* to the *G*, and the *Z* to the *B*.

## 6.3 Registration

### 6.3.1 Overview

A single scan by a structured-light scanner typically provides a range image that covers only part of an object. Therefore, multiple scans from different viewpoints are necessary to capture the entire surface of the object. These multiple range images create a well-known problem called *registration* – aligning all the range images into a common coordinate system. Automatic registration is very difficult since we do not have any prior information about the overall object shape except what is given in each range image, and since finding the correspondence between two range images taken from arbitrary viewpoints is non-trivial.

The Iterative Closest Point (ICP) algorithm [8, 13, 62] made a significant contribution on solving the registration problem. It is an iterative algorithm for registering two data sets. In each iteration, it selects the closest points between two data sets as corresponding points, and computes a rigid transformation that minimizes the distances between corresponding points. The data set is updated by applying the transformation, and the iterations continued until the error between corresponding points falls below a preset threshold. Since the algorithm involves the minimization of mean-square distances, it may converge to a local minimum instead of global minimum. This implies that a good initial registration must be given as a starting point, otherwise the algorithm may converge to a local minimum that is far from the best solution. Therefore, a technique that provides a good initial registration is necessary.

One example for solving the initial registration problem is to attach the scanning system to a robotic arm and keep track of the position and the orientation of the scanning system. Then, the transformation matrices corresponding to the different viewpoints are directly provided. However, such a system requires additional expensive hardware. Also, it requires the object to be stationary, which means that the object cannot be repositioned for the purpose of acquiring data from new viewpoints. Another alternative for solving the initial registration is to design a graphical user interface that allows a human to interact with the data, and perform the registration manually.

Since the ICP algorithm registers two sets of data, another issue that should be considered is registering a set of multiple range data that mini-

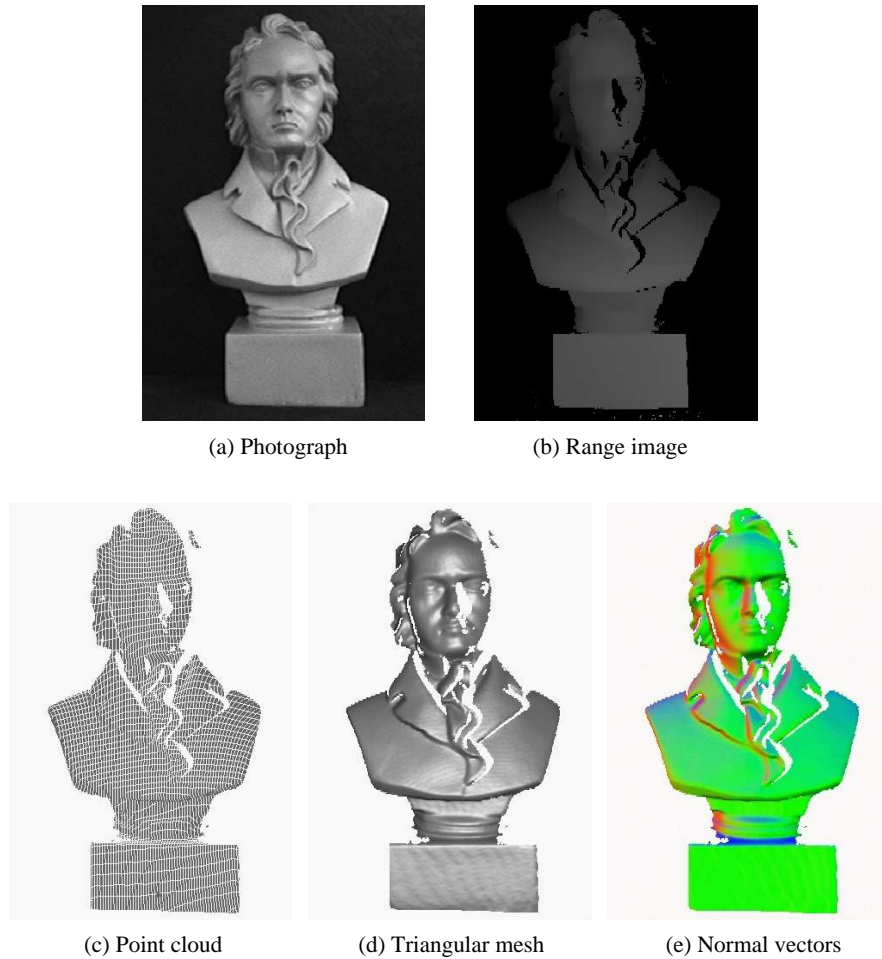


Fig. 6.10: Geometric data acquired by our structured-light scanner  
(a): The photograph of the figure that was scanned. (b): The range image displayed as intensity values. (c): The computed 3D coordinates as point cloud. (d): The shaded triangular mesh. (e): The normal vectors displayed as *RGB* colors where the *X* component of the normal vector corresponds to the *R* component, the *Y* to the *G*, and the *Z* to the *B*.

mizes the registration error between all pairs. This problem is often referred to as *multi-view registration*, and we will discuss in more detail in Section 6.3.5.

### 6.3.2 Iterative Closest Point (ICP) Algorithm

The ICP algorithm was first introduced by Besl and McKay [8], and it has become the principle technique for registration of 3D data sets. The algorithm takes two 3D data sets as input. Let  $\mathbf{P}$  and  $\mathbf{Q}$  be two input data sets containing  $N_p$  and  $N_q$  points respectively. That is,  $\mathbf{P} = \{\mathbf{p}_i\}$ ,  $i = 1, \dots, N_p$ , and  $\mathbf{Q} = \{\mathbf{q}_i\}$ ,  $i = 1, \dots, N_q$ . The goal is to compute a rotation matrix  $\mathbf{R}$  and a translation vector  $\mathbf{t}$  such that the transformed set  $\mathbf{P}' = \mathbf{R}\mathbf{P} + \mathbf{t}$  is best aligned with  $\mathbf{Q}$ . The following is a summary of the algorithm (See Figure 6.11 for a pictorial illustration of the ICP).

1. **Initialization:**  $k = 0$  and  $\mathbf{P}_k = \mathbf{P}$ .
2. **Compute the closest point:** For each point in  $\mathbf{P}_k$ , compute its closest point in  $\mathbf{Q}$ . Consequently, it produces a set of closest points  $\mathbf{C} = \{\mathbf{c}_i\}$ ,  $i = 1, \dots, N_p$  where  $\mathbf{C} \subset \mathbf{Q}$ , and  $\mathbf{c}_i$  is the closest point to  $\mathbf{p}_i$ .
3. **Compute the registration:** Given the set of closest points  $\mathbf{C}$ , the mean square objective function to be minimized is:

$$f(\mathbf{R}, \mathbf{t}) = \frac{1}{N_p} \sum_{i=1}^{N_p} \|\mathbf{c}_i - \mathbf{R}\mathbf{p}_i - \mathbf{t}\|^2 \quad (18)$$

Note that  $\mathbf{p}_i$  is a point from the original set  $\mathbf{P}$ , not  $\mathbf{P}_k$ . Therefore, the computed registration applies to the original data set  $\mathbf{P}$  whereas the closest points are computed using  $\mathbf{P}_k$ .

4. **Apply the registration:**  $\mathbf{P}_{k+1} = \mathbf{R}\mathbf{P} + \mathbf{t}$ .
5. **If the desired precision of the registration is met:** Terminate the iteration.  
**Else:**  $k = k + 1$  and repeat steps 2-5.

Note that the 3D data sets  $\mathbf{P}$  and  $\mathbf{Q}$  do not necessarily need to be points. It can be a set of lines, triangles, or surfaces as long as closest entities can be computed and the transformation can be applied. It is also important to note that the algorithm assumes all the data in  $\mathbf{P}$  lies inside the boundary of  $\mathbf{Q}$ . We will later discuss about relaxing this assumption.

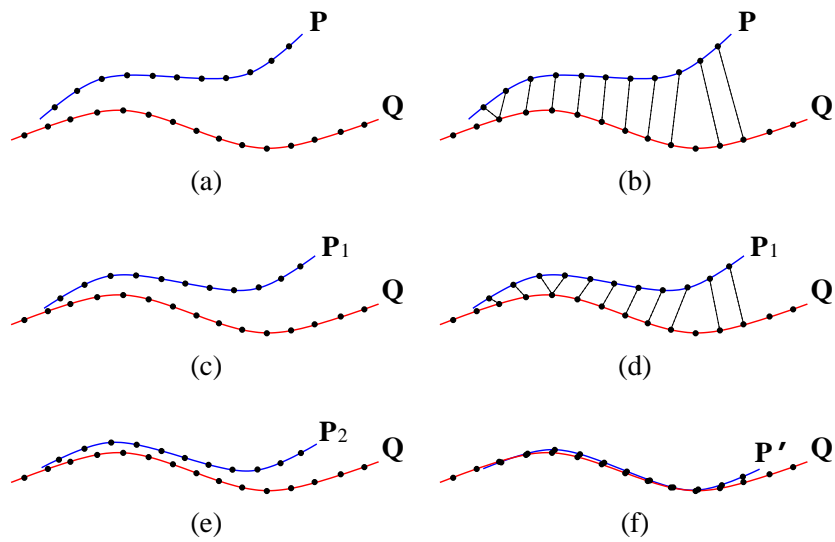


Fig. 6.11: Illustration of the ICP algorithm

(a): Initial  $\mathbf{P}$  and  $\mathbf{Q}$  to register. (b): For each point in  $\mathbf{P}$ , find a corresponding point, which is the closest point in  $\mathbf{Q}$ . (c): Apply  $\mathbf{R}$  and  $\mathbf{t}$  from Eq. (18) to  $\mathbf{P}$ . (d): Find a new corresponding point for each  $\mathbf{P}_1$ . (e): Apply new  $\mathbf{R}$  and  $\mathbf{t}$  that were computed using the new corresponding points. (f): Iterate the process until converges to a local minimum.

Given the set of closest points  $\mathbf{C}$ , the ICP computes the rotation matrix  $\mathbf{R}$  and the translation vector  $\mathbf{t}$  that minimizes the mean square objective function of Eq. (18). Among other techniques, Besl and McKay in their paper chose the solution of Horn [25] using unit quaternions. In that solution, the mean of the closet point set  $\mathbf{C}$  and the mean of the set  $\mathbf{P}$  are respectively given by

$$\mathbf{m}_c = \frac{1}{N_p} \sum_{i=1}^{N_p} \mathbf{c}_i \quad , \quad \mathbf{m}_p = \frac{1}{N_p} \sum_{i=1}^{N_p} \mathbf{p}_i.$$

The new coordinates, which have zero means are given by

$$\mathbf{c}'_i = \mathbf{c}_i - \mathbf{m}_c \quad , \quad \mathbf{p}'_i = \mathbf{p}_i - \mathbf{m}_p.$$

Let a  $3 \times 3$  matrix  $\mathbf{M}$  be given by

$$\begin{aligned} \mathbf{M} &= \sum_{i=1}^{N_p} \mathbf{p}'_i \mathbf{c}'_i{}^T \\ &= \begin{bmatrix} S_{xx} & S_{xy} & S_{xz} \\ S_{yx} & S_{yy} & S_{yz} \\ S_{zx} & S_{zy} & S_{zz} \end{bmatrix}, \end{aligned}$$

which contains all the information required to solve the least squares problem for rotation. Let us construct a  $4 \times 4$  symmetric matrix  $\mathbf{N}$  given by

$$\mathbf{N} = \begin{bmatrix} S_{xx} + S_{yy} + S_{zz} & S_{yz} - S_{zy} & S_{zx} - S_{xz} & S_{xy} - S_{yx} \\ S_{yz} - S_{zy} & S_{xx} - S_{yy} - S_{zz} & S_{xy} + S_{yx} & S_{zx} + S_{xz} \\ S_{zx} - S_{xz} & S_{xy} + S_{yx} & -S_{xx} + S_{yy} - S_{zz} & S_{yz} + S_{zy} \\ S_{xy} - S_{yx} & S_{zx} + S_{xz} & S_{yz} + S_{zy} & -S_{xx} - S_{yy} + S_{zz} \end{bmatrix}$$

Let the eigenvector corresponding to the largest eigenvalue of  $\mathbf{N}$  be  $\mathbf{e} = [e_0 \ e_1 \ e_2 \ e_3]$  where  $e_0 \geq 0$  and  $e_0^2 + e_1^2 + e_2^2 + e_3^2 = 1$ . Then, the rotation matrix  $\mathbf{R}$  is given by

$$\mathbf{R} = \begin{bmatrix} e_0^2 + e_1^2 - e_2^2 - e_3^2 & 2(e_1e_2 - e_0e_3) & 2(e_1e_3 - e_0e_2) \\ 2(e_1e_2 + e_0e_3) & e_0^2 - e_1^2 + e_2^2 - e_3^2 & 2(e_2e_3 - e_0e_1) \\ 2(e_1e_3 - e_0e_2) & 2(e_2e_3 + e_0e_1) & e_0^2 - e_1^2 - e_2^2 + e_3^2 \end{bmatrix}.$$

Once we compute the optimal rotation matrix  $\mathbf{R}$ , the optimal translation vector  $\mathbf{t}$  can be computed by

$$\mathbf{t} = \mathbf{m}_c - \mathbf{R}\mathbf{m}_p.$$

A complete derivation and proofs can be found in [25]. A similar method is also presented in [17].

The convergence of ICP algorithm can be accelerated by extrapolating the registration space. Let  $\mathbf{r}_i$  be a vector that describes a registration (i.e., rotation and translation) at  $i$ th iteration. Then, its direction vector in the registration space is given by

$$\Delta\mathbf{r}_i = \mathbf{r}_i - \mathbf{r}_{i-1}, \quad (19)$$

and the angle between the last two directions is given by

$$\theta_i = \cos^{-1} \left( \frac{\Delta\mathbf{r}_i^T \Delta\mathbf{r}_{i-1}}{\|\Delta\mathbf{r}_i\| \|\Delta\mathbf{r}_{i-1}\|} \right). \quad (20)$$

If both  $\theta_i$  and  $\theta_{i-1}$  are small, then there is a good direction alignment for the last three registration vectors  $\mathbf{r}_i$ ,  $\mathbf{r}_{i-1}$ , and  $\mathbf{r}_{i-2}$ . Extrapolating these three registration vectors using either linear or parabola update, the next registration vector  $\mathbf{r}_{i+1}$  can be computed. They showed 50 iterations of normal ICP was accelerated to about 15 to 20 iterations using such a technique.

### 6.3.3 Variants of ICP

Since the introduction of the ICP algorithm, various modifications have been developed in order to improve its performance .

Chen and Medioni [12, 13] developed a similar algorithm around the same time. The main difference is its strategy for point selection and for finding the correspondence between the two data sets. The algorithm first selects initial points on a regular grid, and computes the local curvature of these points. The algorithm only selects the points on smooth areas, which they call “control points”. Their point selection method is an effort to save computation time, and to have reliable normal directions on the control points. Given the control points on one data set, the algorithm finds the correspondence by computing the intersection between the line that passes through the control point in the direction of its normal and the surface of the other data set. Although the authors did not mention in their paper, the advantage of their method is that the correspondence is less sensitive to noise and to outliers. As illustrated in Fig. 6.12, the original ICP’s correspondence method may select outliers in the data set  $\mathbf{Q}$  as corresponding points since the distance is the only constraint. However, Chen and Medioni’s method is less sensitive to noise since the normal directions of the control points in  $\mathbf{P}$  are reliable, and the noise in  $\mathbf{Q}$  have no effect in finding the correspondence. They also briefly discussed the issues in registering multiple range data (i.e., multi-view registration). When registering multiple range data, instead of registering with a single neighboring range data each time, they suggested to register with the previously registered data as a whole. In this way, the information from all the previously registered data can be used. We will elaborate

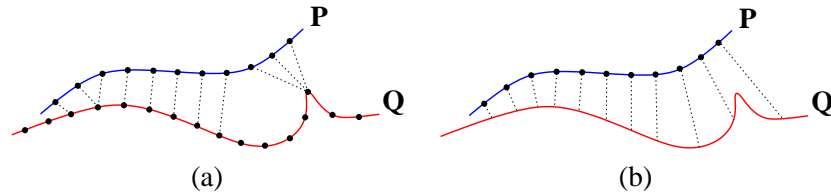


Fig. 6.12: Advantage of Chen and Medioni's algorithm.

- (a): Result of the original ICP's correspondence method in the presence of noise and outliers.  
 (b): Since Chen and Medioni's algorithm uses control points on smooth area and its normal direction, it is less sensitive to noise and outliers.

the discussion in multi-view registration in a separate section later.

Zhang [62] introduced a dynamic thresholding based variant of ICP, which rejects some corresponding points if the distance between the pair is greater than a threshold  $D_{max}$ . The threshold is computed dynamically in each iteration by using statistics of distances between the corresponding points as follows:

```

if  $\mu < D$            /* registration is very good */
     $D_{max} = \mu + 3\sigma$ 
else if  $\mu < 3D$      /* registration is good */
     $D_{max} = \mu + 2\sigma$ 
else if  $\mu < 6D$      /* registration is not good */
     $D_{max} = \mu + \sigma$ 
else                 /* registration is bad */
     $D_{max} = \xi$ 

```

where  $\mu$  and  $\sigma$  are the mean and the standard deviation of distances between the corresponding points.  $D$  is a constant that indicates the expected mean distance of the corresponding points when the registration is good. Finally,  $\xi$  is a maximum tolerance distance value when the registration is bad. This modification relaxed the constraint of the original ICP, which required one data set to be a complete subset of the other data set. As illustrated in Figure 6.13, rejecting some corresponding pairs that are too far apart can lead to a better registration, and more importantly, the algorithm can be applied to partially overlapping data sets. The author also suggested that the points be stored in a k-D tree for efficient closest-point search.

Turk and Levoy [57] added a weight term (i.e., confidence measure) for each 3D point by taking a dot product of the point's normal vector and the

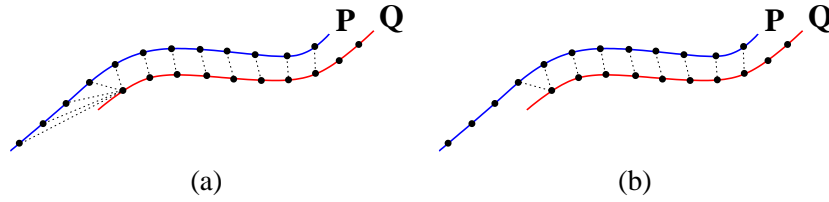


Fig. 6.13: Advantage of Zhang's algorithm.

(a): Since the original ICP assumes  $\mathbf{P}$  is a subset of  $\mathbf{Q}$ , it finds corresponding points for all  $\mathbf{P}$ . (b): Zhang's dynamic thresholding allows  $\mathbf{P}$  and  $\mathbf{Q}$  to be partially overlapping.

vector pointing to the light source of the scanner. This was motivated by the fact that structured-light scanning acquires more reliable data when the object surface is perpendicular to the laser plane. Assigning lower weights to unreliable 3D points (i.e., points on the object surface nearly parallel with the laser plane) helps to achieve a more accurate registration. The weight of a corresponding pair is computed by multiplying the weights of the two corresponding points. Let the weights of corresponding pairs be  $\mathbf{w} = \{w_i\}$ , then the objective function in Eq. (18) is now a weighted function:

$$f(\mathbf{R}, \mathbf{t}) = \frac{1}{N_p} \sum_{i=1}^{N_p} w_i \|\mathbf{c}_i - \mathbf{R}\mathbf{p}_i - \mathbf{t}\|^2 \quad (21)$$

For faster and efficient registration, they proposed to use increasingly more detailed data from a hierarchy during the registration process where less detailed data are constructed by sub-sampling range data. Their modified ICP starts with the lowest-level data, and uses the resulting transformation as the initial position for the next data in the hierarchy. The distance threshold is set as twice of sampling resolution of current data. They also discarded corresponding pairs in which either points is on a boundary in order to make reliable correspondences.

Masuda *et al.* [38, 37] proposed an interesting technique in an effort to add robustness to the original ICP. The motivation of their technique came from the fact that a local minimum obtained by the ICP algorithm is predicted by several factors such as initial registration, selected points and corresponding pairs in the ICP iterations, and that the outcome would be more unpredictable when noise and outliers exist in the data. Their algorithm consists of two main stages. In the first stage, the algorithm performs the ICP a number of times, but in each trial the points used for ICP calculations are selected differently based on random sampling. In the second stage, the algorithm selects the transformation that produced the minimum median distance between the corresponding pairs as the final resulting transformation. Since

the algorithm performs the ICP a number of times with differently selected points, and chooses the best transformation, it is more robust especially with noise and outliers.

Johnson and Kang [29] introduced “color ICP” technique in which the color information is incorporated along with the shape information in the closest-point (i.e., correspondence) computation. The distance metric  $d$  between two points  $\mathbf{p}$  and  $\mathbf{q}$  with the 3D location and the color are denoted as  $(x, y, z)$  and  $(r, g, b)$  respectively can be computed as

$$d^2(\mathbf{p}, \mathbf{q}) = d_e^2(\mathbf{p}, \mathbf{q}) + d_c^2(\mathbf{p}, \mathbf{q}) \quad (22)$$

where

$$d_e(\mathbf{p}, \mathbf{q}) = \sqrt{(x_p - x_q)^2 + (y_p - y_q)^2 + (z_p - z_q)^2}, \quad (23)$$

$$d_c(\mathbf{p}, \mathbf{q}) = \sqrt{\lambda_1(r_p - r_q)^2 + \lambda_2(g_p - g_q)^2 + \lambda_3(b_p - b_q)^2} \quad (24)$$

and  $\lambda_1, \lambda_2, \lambda_3$  are constants that control the relative importance of the different color components and the importance of color overall vis-a-vis shape. The authors have not discussed how to assign values to the constants, nor the effect of the constants on the registration. A similar method was also presented in [21].

Other techniques employ using other attributes of a point such as normal direction [53], curvature sign classes [19], or combination of multiple attributes [50], and these attributes are combined with the Euclidean distance in searching for the closest point. Following these works, Godin *et al.* [20] recently proposed a method for the registration of attributed range data based on a random sampling scheme. Their random sampling scheme differs from that of [38, 37] in that it uses the distribution of attributes as a guide for point selection as opposed to uniform sampling used in [38, 37]. Also, they use attribute values to construct a compatibility measure for the closest point search. That is, the attributes serve as a boolean operator to either accept or reject a correspondence between two data points. This way, the difficulty of choosing constants in distance metric computation, for example  $\lambda_1, \lambda_2, \lambda_3$  in Eq. (24), can be avoided. However, a threshold for accepting and rejecting correspondences is still required.

### 6.3.4 Initial Registration

Given two data sets to register, the ICP algorithm converges to different local minima depending on the initial positions of the data sets. Therefore, it is not guaranteed that the ICP algorithm will converge to the desired global minimum, and the only way to confirm the global minimum is to find the minimum of all the local minima. This is a fundamental limitation of the ICP that it requires a good initial registration as a starting point to maxi-

mize the probability of converging to a correct registration. Besl and McKay in their ICP paper [8] suggested to use a set of initial registrations chosen by sampling of quaternion states and translation vector. If some geometric properties such as principle components of the data sets provide distinctness, such information may be used to help reduce the search space.

As mentioned before, one can provide initial registrations by a tracking system that provides relative positions of each scanning viewpoint. One can also provide initial registrations manually through human interaction. Some researchers have proposed other techniques for providing initial registrations [11, 17, 22, 28], but it is reported in [46] that these methods do not work reliably for arbitrary data.

Recently, Huber [26] proposed an automatic registration method in which no knowledge of data sets is required. The method constructs a globally consistent model from a set of pairwise registration results. Although the experiments showed good results considering the fact that the method does not require any initial information, there was still some cases where incorrect registration was occurred.

### 6.3.5 Multi-view Registration

Although the techniques we have reviewed so far only deal with pairwise registration – registering two data sets, they can easily be extended to multi-view registration – registering multiple range images while minimizing the registration error between all possible pairs. One simple and obvious way is to perform a pairwise registration for each of two neighboring range images sequentially. This approach, however, accumulates the errors from each registration, and may likely have a large error between the first and the last range image.

Chen and Medioni [13] were the first to address the issues in multi-view registration. Their multi-view registration goes as follows: First, a pairwise registration between two neighboring range images is carried out. The resulting registered data is called a meta-view. Then, another registration between a new unregistered range image and the meta-view is performed, and the new data is added to the meta-view after the registration. This process is continued until all range images are registered. The main drawback of the meta-view approach is that the newly added images to the meta-view may contain information that could have improved the registrations performed previously.

Bergevin *et al.* [5, 18] noticed this problem, and proposed a new method that considers the network of views as a whole and minimizes the registration errors for all views simultaneously. Given  $N$  range images from the viewpoints  $V_1, V_2, \dots, V_N$ , they construct a network such that  $N - 1$  viewpoints are linked to one central viewpoint in which the reference coordinate system

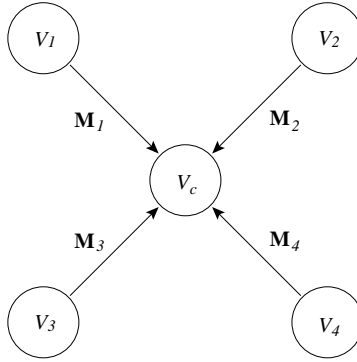


Fig. 6.14: Network of multiple range data was considered in the multi-view registration method by Bergevin *et al.* [5, 18]

is defined. For each link, an initial transformation matrix  $\mathbf{M}_{i,0}$  that brings the coordinate system of  $V_i$  to the reference coordinate system is given. For example, consider the case of 5 range images shown in Fig. 6.14 where viewpoints  $V_1$  through  $V_4$  are linked to a central viewpoint  $V_c$ . During the algorithm, 4 incremental transformation matrices  $\mathbf{M}_{1,k}, \dots, \mathbf{M}_{4,k}$  are computed in each iteration  $k$ . In computing  $\mathbf{M}_{1,k}$ , range images from  $V_2, V_3$  and  $V_4$  are transformed to the coordinate system of  $V_1$  by first applying its associated matrix  $\mathbf{M}_{i,k-1}$ ,  $i = 2, 3, 4$  followed by  $\mathbf{M}_{1,k-1}^{-1}$ . Then, it computes the corresponding points between the range image from  $V_1$  and the three transformed range images.  $\mathbf{M}_{1,k}$  is the transformation matrix that minimizes the distances of all the corresponding points for all the range images in the reference coordinate system. Similarly,  $\mathbf{M}_{2,k}, \mathbf{M}_{3,k}$  and  $\mathbf{M}_{4,k}$  are computed, and all these matrices are applied to the associated range images simultaneously at the end of iteration  $k$ . The iteration continues until all the incremental matrices  $\mathbf{M}_{i,k}$  become close to identity matrices.

Benjemaa and Schmitt [4] accelerated the above method by applying each incremental transformation matrix  $\mathbf{M}_{i,k}$  immediately after it is computed instead of applying all simultaneously at the end of the iteration. In order to not favor any individual range image, they randomized the order of registration in each iteration.

Pulli [45, 46] argued that these methods cannot easily be applied to large data sets since they require large memory to store all the data, and since the methods are computationally expensive as  $N - 1$  ICP registrations are performed. To get around these limitations, his method first performs pairwise registrations between all neighboring views that result in overlapping range images. The corresponding points discovered in this manner are used in

the next step that does multi-view registration. The multi-view registration process is similar to that of Chen and Medioni except for the fact that the corresponding points from the previous pairwise registration step are used as permanent corresponding points throughout the process. Thus, searching for corresponding points, which is computationally most demanding, is avoided, and the process does not require large memory to store all the data. The author claimed that his method, while being faster and less demanding on memory, results in similar or better registration accuracy compared to the previous methods.

### 6.3.6 Experimental Result

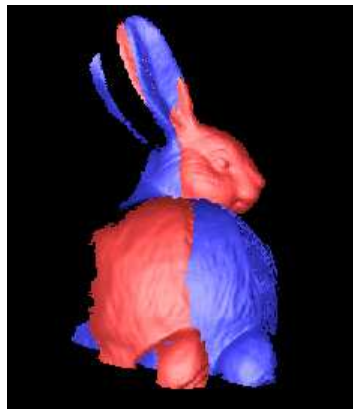
We have implemented a modified ICP algorithm for registration of our range images. Our algorithm uses Zhang's dynamic thresholding for rejecting correspondences. In each iteration, a threshold  $D_{max}$  is computed as

$$D_{max} = m + 3\sigma$$

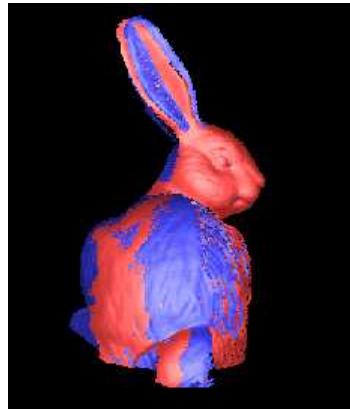
where  $m$  and  $\sigma$  are the mean and the standard deviation of the distances of the corresponding points. If the Euclidean distance between two corresponding points exceeds this threshold, the correspondence is rejected. Our algorithm also uses the bucketing algorithm (i.e., Elias algorithm) for fast corresponding point search. Figure 6.15 shows an example of a pairwise registration. Even though the initial positions were relatively far from the correct registration, it successfully converged in 53 iterations. Notice in the final result (Figure 6.15(d)) that the overlapping surfaces are displayed with many small patches, which indicates that the two data sets are well registered.

We acquired 40 individual range images from different viewpoints to capture the entire surface of the bunny figure. Twenty range images covered about 90% of the entire surface. The remaining 10% of surface was harder to view on account of either self-occlusions or because the object would need to be propped so that those surfaces would become visible to the sensor. Additional 20 range images were gathered to get data on such surfaces.

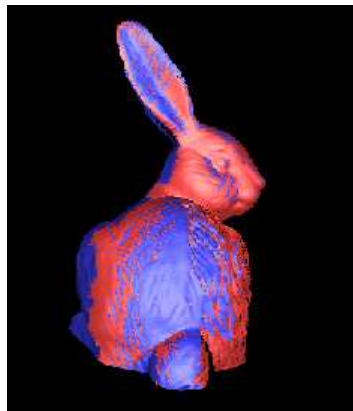
Our registration process consists of two stages. In the first stage, it performs a pairwise registration between a new range image and all the previous range images that are already registered. When the new range image's initial registration is not available, for example when the object is repositioned, it first goes through a human assisted registration process that allows a user to visualize the new range image in relation to the previously registered range images. The human is able to rotate one range image vis-a-vis the other and provide corresponding points. See Figure 6.16 for an illustration of the human assisted registration process. The corresponding points given by the human are used to compute an initial registration for the new range image. Subsequently, registration proceeds as before.



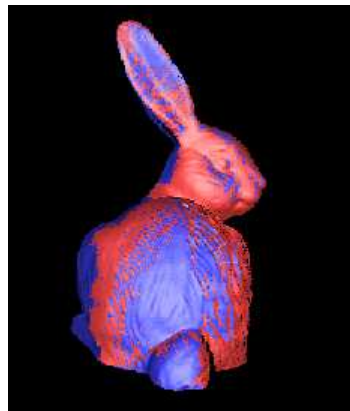
(a) Initial positions



(b) After 20 iterations



(c) After 40 iterations



(d) Final after 53 iterations

Fig. 6.15: Example of a pairwise registration using the ICP algorithm

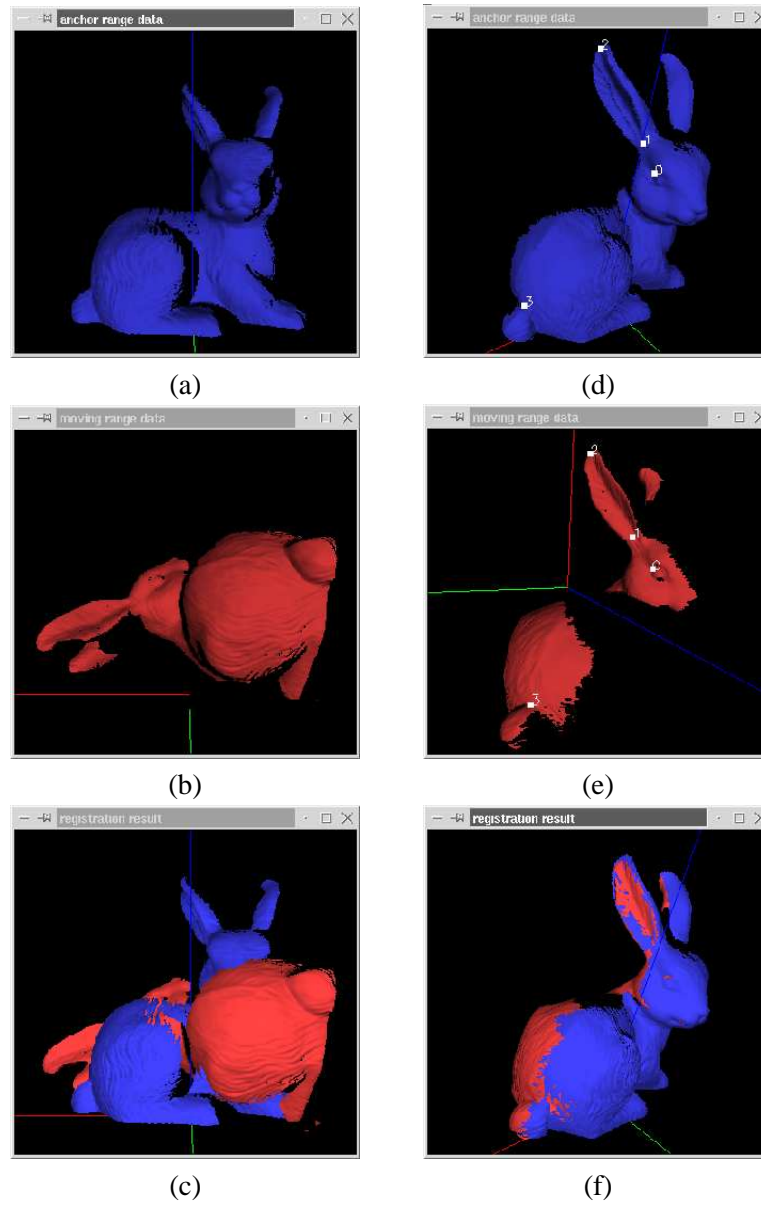


Fig. 6.16: Human assisted registration process

(a),(b),(c): Initial Positions of two data sets to register. (d),(e): User can move around the data and click corresponding points. (f): The given corresponding points are used to compute an initial registration.

Registration of all the range images in the manner described above constitutes the first stage of the overall registration process. The second stage then fine-tunes the registration by performing a multi-view registration using the method presented in [4]. Figure 6.17 shows the 40 range images after the second stage.

## 6.4 Integration

Successful registration aligns all the range images into a common coordinate system. However, the registered range images taken from adjacent view-points will typically contain overlapping surfaces with common features in the areas of overlap. The integration process eliminates the redundancies, and generates a single connected surface model.

Integration methods can be divided into five different categories: volumetric method, mesh stitching method, region-growing method, projection method, and sculpting-based method. In the next sub-sections we will explain each of these categories.

### 6.4.1 Volumetric Methods

The volumetric method consists of two stages. In the first stage, an implicit function  $d(\mathbf{x})$  that represents the closest distance from an arbitrary point  $\mathbf{x} \in \mathbb{R}^3$  to the surface we want to reconstruct is computed. Then the object surface can be represented by the equation  $d(\mathbf{x}) = 0$ . The sign of  $d(\mathbf{x})$  indicates whether  $\mathbf{x}$  lies outside or inside the surface. In the second stage, the isosurface – the surface defined by  $d(\mathbf{x}) = 0$  – is extracted by triangulating the zero-crossing points of  $d(\mathbf{x})$  using the marching cubes algorithm [36, 39]. The most important task here is to reliably compute the function  $d(\mathbf{x})$  such that this function best approximates the true surface of the object. Once  $d(\mathbf{x})$  is approximated, other than the marching cubes algorithm, such as marching triangles algorithm, can be used to extract the isosurface.

The basic concept of the volumetric method is illustrated in Figure 6.18. First, a 3D volumetric grid that contains the entire surface is generated, and all the cubic cells (or voxels) are initialized as “empty”. If the surface is found “near” the voxel (the notion of “near” will be defined later), the voxel is set to “non-empty” and  $d(\mathbf{x})$  for each of the 8 vertices of the voxel is computed by the signed distance between the vertex to the closest surface point. The sign of  $d(\mathbf{x})$  is positive if the vertex is outside the surface, and negative otherwise. After all the voxels in the grid are tested, the triangulation is performed as follows. For each non-empty voxel, zero crossing points of  $d(\mathbf{x})$ , if any, are computed. The computed zero crossing points are then triangulated.

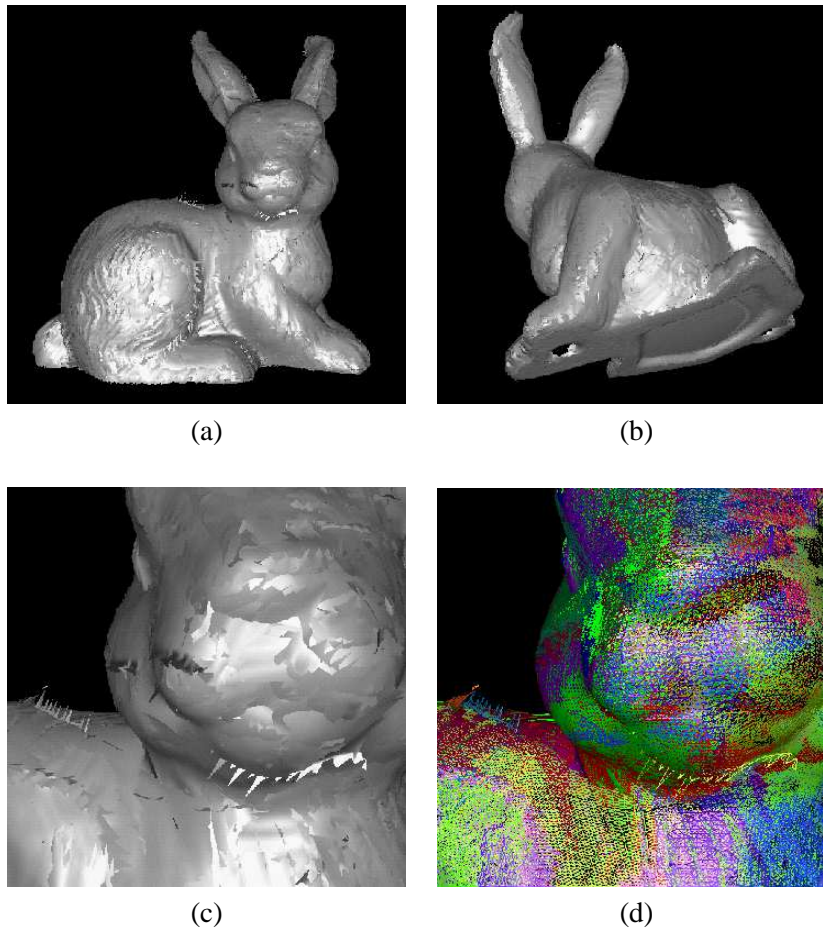


Fig. 6.17: 40 range images after the second stage of the registration process (a),(b): Two different views of the registered range images. All the range images are displayed as shaded triangular mesh. (c): Close-up view of the registered range images. (d): The same view as (c) displayed with triangular edges. Each color represents an individual range image.

lated by applying one of the 15 cases in the marching cubes look-up-table.<sup>1</sup> For example, the upper-left voxel in Figure 6.18(d) corresponds to the case number 2 of the look-up-table, and the upper-right and the lower-left voxels both correspond to the case number 8. Triangulating zero crossing points of all the non-empty voxels results the approximated isosurface.

We will now review three volumetric methods. The main difference between these three methods lies in how the implicit function  $d(\mathbf{x})$  is computed.

Curless and Levoy [14] proposed a technique tuned for range images generated by a structured-light scanner. Suppose we want to integrate  $n$  range images where all the range images are in the form of triangular mesh. For each range image  $i$ , two functions  $d_i(\mathbf{x})$  and  $w_i(\mathbf{x})$  are computed where  $d_i(\mathbf{x})$  is the signed distance from  $\mathbf{x}$  to the nearest surface along the viewing direction of the  $i$ th range image and  $w_i(\mathbf{x})$  is the weight computed by interpolating the three vertices of the intersecting triangle (See Figure 6.19). The weight of each vertex is computed as the dot product between the normal direction of the vertex and the viewing direction of the sensor. Additionally, lower weights are assigned to the vertices that are near a surface discontinuity. After processing all the range images,  $d(\mathbf{x})$  is constructed by combining  $d_i(\mathbf{x})$  and the associated weight function  $w_i(\mathbf{x})$  obtained from the  $i$ th range image. That is,

$$d(\mathbf{x}) = \frac{\sum_{i=1}^n w_i(\mathbf{x})d_i(\mathbf{x})}{\sum_{i=1}^n w_i(\mathbf{x})}$$

We said earlier that  $d(\mathbf{x})$  is computed at the vertices of a voxel if the surface is “near”. In other words,  $d(\mathbf{x})$  is sampled only if the distance between the vertex to the nearest surface point is less than some threshold. Without imposing this threshold, computing and storing  $d(\mathbf{x})$  for all the voxels in each range image will be impractical, but more importantly, the surfaces on opposite sides will interfere with each other since the final  $d(\mathbf{x})$  is the weighted average of  $d_i(\mathbf{x})$  obtained from  $n$  range images. Therefore, the threshold must be small enough to avoid the interference between the surfaces on opposite sides, but large enough to acquire multiple samples of  $d_i(\mathbf{x})$  that will contribute to a reliable computation of  $d(\mathbf{x})$  and subsequent zero crossing points. Considering this tradeoff, a practical suggestion would be to set the threshold as half the maximum uncertainty of the range measurement.

Hoppe *et al.* [24] were the first to propose the volumetric method. Their algorithm is significant in that it assumes the input data is unorganized. That is, neither the connectivity nor the normal direction of points is known in advance. Therefore, the method first estimates the oriented tangent plane for each data point. The tangent plane is computed by fitting the best plane in

---

<sup>1</sup>Since there are 8 vertices in a voxel, there are 256 ways in which the surface can intersect the voxel. These 256 cases can be reduced to 15 general cases by applying the reversal symmetry and the rotational symmetry.

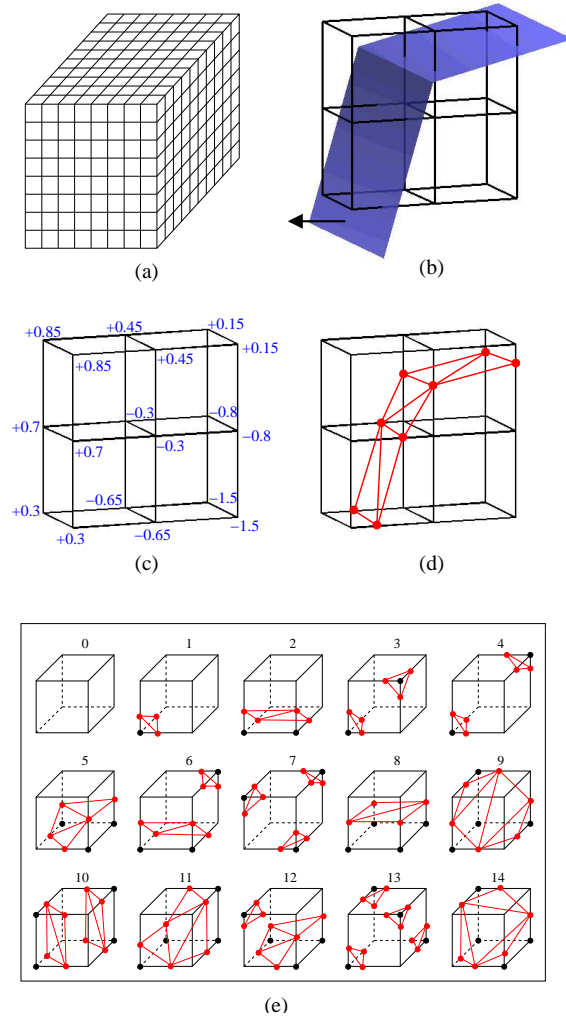


Fig. 6.18: Volumetric Method

(a): 3D volumetric grid. (b): Four neighboring cubes near the surface. The arrow points to the outside of the surface. (c): Signed distance function  $d(\mathbf{x})$  is sampled at each vertex. (d): Zero-crossing points of  $d(\mathbf{x})$  (red circles) are triangulated by the marching cubes algorithm. (e): 15 general cases of the marching cubes algorithm.

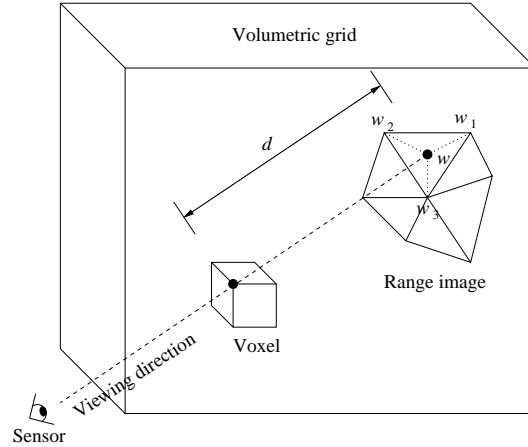


Fig. 6.19: Computing  $d(\mathbf{x})$  in Curless and Levoy's method [14]

the least squares sense on  $k$  nearest neighbors. Then,  $d(\mathbf{x})$  is the distance between  $\mathbf{x}$  and its closest point's tangent plane.

Wheeler *et al.* [59] proposed a similar method called "consensus-surface algorithm". Their algorithm emphasizes the selection of points used to compute the signed-distance in order to deal with noise and outliers.

#### 6.4.2 Mesh Stitching Methods

The Mesh stitching method was first introduced by Soucy and Laurendeau [51, 52]. Their method consists of three main steps: (1) determining redundant surface regions, (2) reparameterizing those regions into non-redundant surface regions, and (3) connecting (or stitching) all the non-redundant surface regions.

Redundant surface regions represent common surface regions sampled by two or more range images. The content of each of the redundant surface regions can be determined by finding all possible pairs of range images and their redundant surface regions. For example, consider Figure 6.20 where 3 range images  $V_1$ ,  $V_2$  and  $V_3$  have 4 different redundant surface regions. If we find the pairwise redundant surface regions of  $V_1V_2$ ,  $V_1V_3$ , and  $V_2V_3$ , it is possible to determine for each point, which range images have sampled that point. Therefore, the contents of 4 redundant surface regions are implicitly available.

Now, we will describe how the redundant surface region between a pair of range images, say  $V_1$  and  $V_2$ , can be found. Two conditions are imposed to determine if a point in  $V_1$  is redundant with  $V_2$ : First, the point must be near

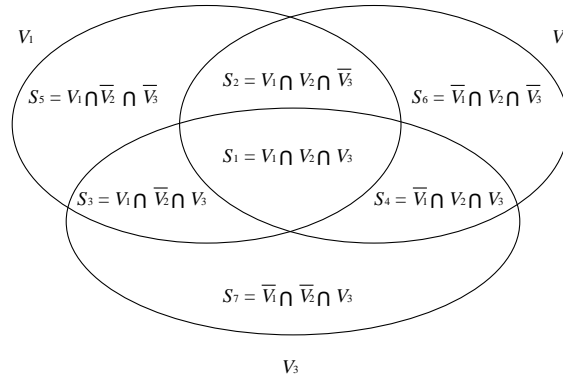


Fig. 6.20: Redundant surfaces of three different range images

the surface of  $V_2$ , and second, the point must be visible from the viewing direction of  $V_2$ . The Spatial Neighborhood Test (SNT), which tests the first condition, checks whether the distance between the point and the surface of  $V_2$  is within the uncertainty of the range sensor. The Surface Visibility Test (SVT), which tests the second condition, checks if the dot product between the normal direction of the point and the viewing direction (i.e., optical axis) of the  $V_2$  is positive. All the points in  $V_1$  that satisfy the two tests are assumed to be in the redundant surface with  $V_2$ . Unfortunately, the SNT and the SVT yield unreliable results in the regions where surface discontinuities occur or when noise is present. Therefore, a heuristic region-growing technique that fine-tunes the estimated redundant surfaces is used. By observing that the boundaries of the redundant surface correspond to the surface discontinuity at least in one of the range images, each of the estimated redundant regions is expanded until it reaches the surface discontinuity of one of the range images. In order to prevent small isolated regions to grow freely, an additional constraint that the expanded region must contain at least 50 percent of the original seed region is imposed.

After the redundant surface regions are determined, those regions are reparameterized into non-redundant surfaces. For each redundant surface region, a plane grid is defined; the plane grid has the same sampling resolution as that of a range image, and passes through the center of mass of the redundant surface region with the normal direction given by the average normal of all the points in the region. All the points in the region are then projected onto this plane grid. Associated with each vertex in the grid is the average of the perpendicular coordinate values for all the points that projected onto the cell represented by that vertex. The grid coordinates together with the computed perpendicular coordinates define new non-redundant surface points that are

then triangulated. After reparameterizing all surface regions, a process that eliminates any remaining overlapping triangles in the boundary of surface regions is performed.

Finally, the non-redundant surface regions obtained in this manner are stitched together by interpolating empty space between the non-redundant surfaces. The interpolation of empty space is obtained by the constrained 2D Delaunay triangulation on the range image grid that sampled that particular empty space continuously. The result after interpolating all the empty spaces is the final connected surface model.

Turk and Levoy [57] proposed a similar method called “mesh zippering”. The main difference between the two algorithms is the order of determining the connectivity and the geometry. The previous algorithm first determines the geometry by reparameterizing the projected points on the grid, then determines the connectivity by interpolating into the empty spaces between the re-parameterized regions. By contrast, Turk and Levoy’s algorithm first determines the connectivity by removing the overlapping surfaces and stitching (or zippering) the borders. Then, it determines the geometry by adjusting surface points as weighted averages of all the overlapping surface points. The mesh zippering algorithm is claimed to be less sensitive to the artifacts of the stitching process since the algorithm first determines the connectivity followed by the geometry.

Let us describe the mesh zippering method in more detail with the illustrations in Figure 6.21. In (a), two partially overlapping surfaces are shown as red and blue triangles. From (b) to (d), the redundant triangles shown as green triangles are removed one by one from each surface until both surfaces remain unchanged. A triangle is redundant if all three distances between its vertices to the other surface are less than a predefined threshold where the threshold is typically set to a small multiple of the range image resolution. After removing the redundant triangles, it finds the boundary edges of one of the two surfaces; the boundary edges of the blue triangles are shown as green lines in (e). Then, the intersections between these boundary edges and the other surface are determined; the intersecting points are depicted as black circles in (f). Since it is unlikely that the boundary edges will exactly intersect the surface, a “thickened wall” is created for each boundary edge; a thickened wall is made of four triangles, and it is locally perpendicular to the boundary edge points of one of the surfaces. The problem now becomes finding intersecting points between the boundary edge wall and the surface. From this point, all the red triangle edges that are beyond the boundary edges are discarded as shown in (g). In (h), the intersecting points are added as new vertices, and triangulated through a constrained triangulation routine [6]. After zippering all the surfaces together, the final step fine-tunes the geometry by considering all the information of the surfaces including those that were discarded in the zippering process. The final position of each surface point

is computed as the weighted average of all the overlapping surfaces along the normal direction of the point. The weight of each point is computed as a dot product between the normal direction of the point and its corresponding range image's viewing direction.

### 6.4.3 Region-Growing Methods

We introduce two region-growing based integration methods. The first method [23], called “marching triangles” consists of two stages. In the first stage, similar to the volumetric method, it defines an implicit surface representation as the zero crossings of a function  $d(\mathbf{x})$ , which defines the signed distance to the nearest point on the surface for any point  $\mathbf{x}$  in 3D space. In the second stage, instead of using the marching cubes algorithm, the marching triangles algorithm is used to triangulate the zero crossings of  $d(\mathbf{x})$ . The marching triangles algorithm starts with a seed triangle, adds a neighbor triangle based on the 3D Delaunay surface constraint, and continues the process until all the points have been considered.

The second method [7], which is more recently developed, is called “ball-pivoting algorithm (BPA)”. The basic principle of the algorithm is that three points form a triangle if a ball of a certain radius  $\rho$  touches all of them without containing any other points. Starting with a seed triangle, the ball pivots around an edge until it touches another point, then forms a new triangle. The process continues until all points have been considered. The BPA is related to the  $\alpha$ -shape<sup>2</sup> [16], thus provides a theoretical guarantee to reconstruct a surface homeomorphic to the original surface within a bounded distance if sufficiently dense and uniform sampling points are given. It is also shown that the BPA can be applied to a large set of data proving that it is efficient in computation and memory usage. The main disadvantage of this method is that the size of radius  $\rho$  must be given manually, and a combination of multiple processes with different  $\rho$  values may be necessary to generate a correct integrated model.

---

2

The  $\alpha$ -shape of a finite point set  $S$  is a polytope uniquely determined by  $S$  and the parameter  $\alpha$  that controls the level-of-detail. A subset  $T \subseteq S$  of size  $|T| = k + 1$  with  $0 \leq k \leq 2$  belongs to a set  $F_{k,\alpha}$  if a sphere of radius  $\alpha$  contains  $T$  without containing any other points in  $S$ . The  $\alpha$ -shape is described by the polytope whose boundary consists of the triangles connecting the points in  $F_{2,\alpha}$ , the edges in  $F_{1,\alpha}$ , and vertices in  $F_{0,\alpha}$ . If  $\alpha = \infty$ , the  $\alpha$ -shape is identical to the convex hull of  $S$ , and if  $\alpha = 0$ , the  $\alpha$ -shape is the point set  $S$  itself.

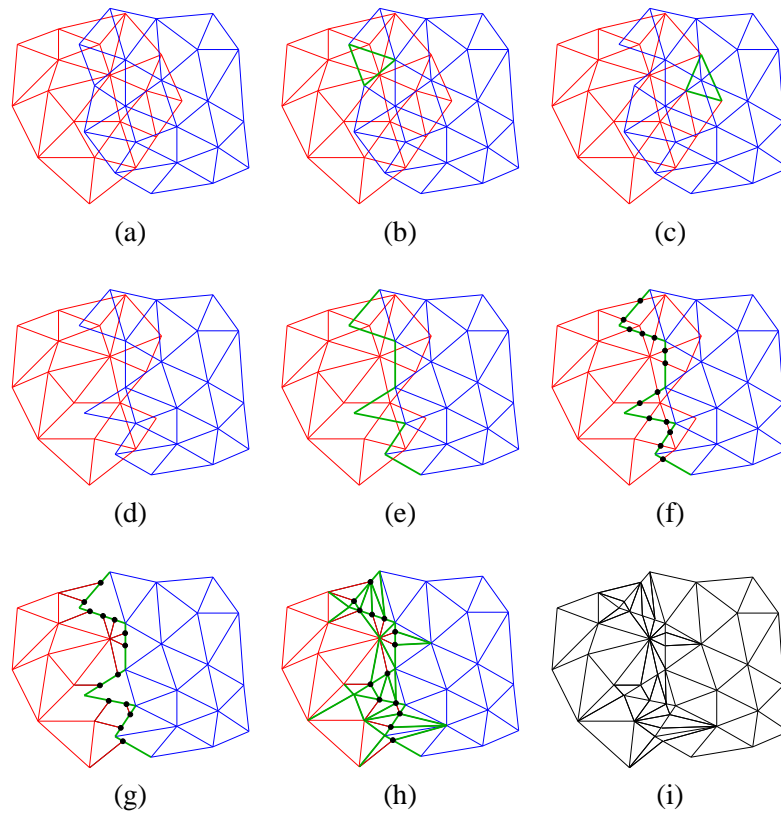


Fig. 6.21: Mesh Zippering algorithm

(a): Two overlapping surfaces. (b): A redundant triangle from the blue surface is removed. (c): A redundant triangle from the red surface is removed. (d): Steps (b) and (c) are continued until both surfaces remain unchanged. (e): After removing all the redundant triangles, the boundary edges blue surface is found. (f): The intersections between the boundary edge and the edges from the red surface are determined. (g): All the edges from the red surface that are beyond the boundary edges are discarded. (h): The intersecting points are added as new vertices and are triangulated. (i): The final position of each point is adjusted by considering all the surfaces including those that were discarded during the zippering process.

#### 6.4.4 Projection Methods

The Projection method [13, 44, 58], one of the earlier integration methods, simply projects the data onto a cylindrical or a spherical grid. Multiple data projections onto a same grid are averaged, and the resulting data is reparameterized. Although this method provides a simple way of integration, it suffers from the fundamental limitation that it can only handle convex objects.

#### 6.4.5 Sculpting Based Methods

The Sculpting based method [1, 2, 10, 16] typically computes tetrahedra volumes from the data points by the 3D Delaunay triangulation. Then, it progressively eliminates tetrahedra until the original shape is extracted. Since the method is based on the Delaunay triangulation, it guarantees that the resulting surface is topologically correct as long as the data points are dense and uniformly distributed. Also, it can be applied to a set of unorganized points. However, the method has difficulty with constructing sharp edges, and it suffers from the expensive computations needed for calculating 3D Delaunay triangulations.

#### 6.4.6 Experimental Result

In order to illustrate the results from integration, let's take as example the Curless and Levoy's volumetric integration method[14]. For that, we used 40 range images of a bunny figurine that were acquired by our structured-light scanning system. All the range images were registered as described in the previous chapter. The integration was performed at 0.5mm resolution (i.e., the size of a voxel of the grid is 0.5mm), which is an approximate sampling resolution of each range image.

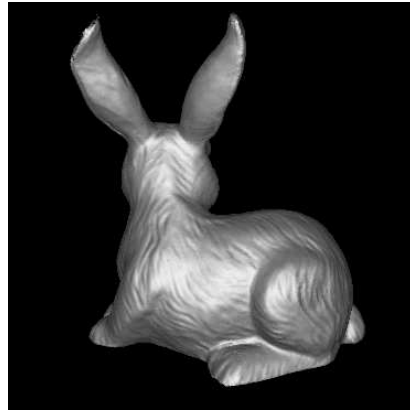
Figure 6.22 shows four different views of the resulting model. The total number of points and triangles in the 40 range images were 1,601,563 and 3,053,907, respectively, and these were reduced to 148,311 and 296,211 in the integrated model. Figure 6.23(a) shows a close-up view around the bunny's nose area before the integration where different colored triangles represent different range images. Figure 6.23(b) shows the same view after the integration.

### 6.5 Acquisition of Reflectance Data

the Successful integration of all range images results in a complete geometric model of an object. This model itself can be the final output if only the shape of the object is desired. But since a photometrically correct visualization is



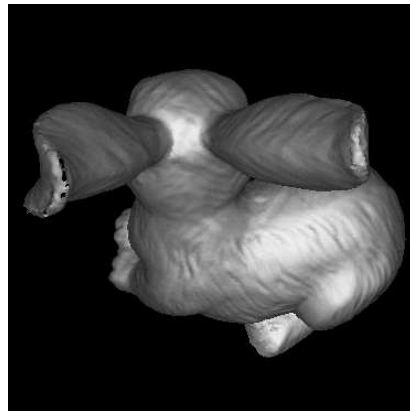
(a)



(b)



(c)



(d)

Fig. 6.22: Integrated model visualized from four different viewpoints

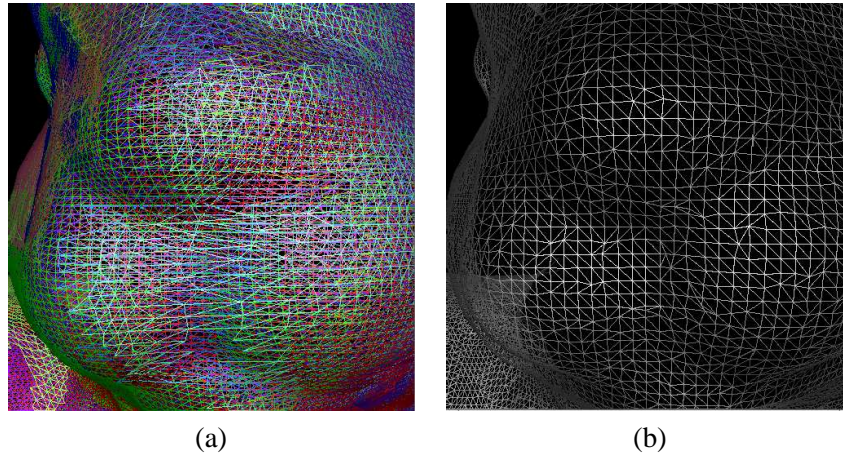


Fig. 6.23: Close-up view of the model before and after integration

frequently required, we must also obtain the reflectance data of the object surface.

In general, there are two approaches for acquiring the reflectance data. The first approach employs one of the many parametric reflectance models and estimates the reflectance parameters for each data point by using multiple images taken from different viewpoints and under different lighting conditions [27, 30, 33, 48, 49]. Once the reflectance parameters are estimated, it is possible to visualize the object under any novel lighting condition from any novel viewpoint. We will describe this approach in greater depth in Section 6.5.2.

The second approach, instead of using a parametric reflectance model, utilizes only a set of color images of the object. Some methods [15, 47] exploit the use of view dependent texture maps. For each viewing direction of the 3D model, a synthetic image for texture mapping is generated by interpolating the input images that were taken from the directions close to the current viewing direction. The synthetic image simulates what would have been the image taken from the current viewing direction, thus it provides a correct texture to the 3D model. Other methods [42, 60] store a series of  $N$  textures for each triangle where the textures are obtained from the color images taken from different viewpoints under known light source directions. The  $N$  textures are compressed by applying the Principal Components Analysis, and a smaller number of textures that approximate basis functions of the viewing space are computed. These basis functions are then interpolated to represent the texture of each triangle from a novel viewpoint.

Although the second approach provides realistic visualization from an

arbitrary viewpoint without estimating reflectance parameters for each data point, one of the major drawbacks is the fact that it can only render the object under the same lighting condition in which the input images were taken. On the other hand, the first approach provides the underlying reflectance properties of the object surface, and thus makes it possible to visualize the object under a novel lighting condition. We will first describe some of the well known reflectance models that are commonly used followed by the methods for estimating reflectance parameters.

### 6.5.1 Reflectance Models

The true reflectance property of an object is based on many complex physical interactions of light with object materials. The Bidirectional Reflectance Distribution Function (BRDF) developed by Nicodemus *et al.* [41] provides a general mathematical function for describing the reflection property of a surface as a function of illumination direction, viewing direction, surface normal, and spectral composition of the illumination used. For our application, we can use the following definition for each of the primary color components:

$$f_r(\theta_i, \phi_i; \theta_r, \phi_r) = \frac{dL_r(\theta_r, \phi_r)}{dE_i(\theta_i, \phi_i)} \quad (25)$$

where  $L_r$  is reflected radiance,  $E_i$  is incident irradiance,  $\theta_i$  and  $\phi_i$  specify the incident light direction, and  $\theta_r$  and  $\phi_r$  specify the reflected direction.

Many researchers have proposed various parametric models to represent the BRDF, each having different strengths and weaknesses. Two of the well known models are those developed by Beckmann and Spizzichino [3], and Torrance and Sparrow [54]. The Beckmann-Spizzichino model was derived using basic concepts of electromagnetic wave theory, and is more general than the Torrance-Sparrow model in the sense that it describes the reflection from smooth to rough surfaces. The Torrance-Sparrow model was developed to approximate reflectance on rough surfaces by geometrically analyzing a path of light ray on rough surfaces. The Torrance-Sparrow model, in general, is more widely used than the Beckman-Spizzichino model because of its simpler mathematical formula.

#### **Torrance-Sparrow Model**

The Torrance-Sparrow model assumes that a surface is a collection of planar micro-facets as shown in Figure 6.24. An infinitesimal surface patch  $dA$  consists of a large set of micro-facets where each facet is assumed to be one side of a symmetric V-cavity. The set of micro-facets has a mean normal vector of  $\mathbf{n}$ , and a random variable  $\alpha$  is used to represent the angle between each micro-facet's normal vector and the mean normal vector. Assuming

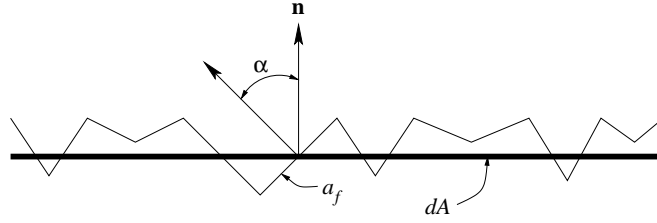


Fig. 6.24: Surface model

the surface patch is isotropic (i.e., rotationally symmetric about surface normal), the distribution of  $\alpha$  can be expressed as a one-dimensional Gaussian distribution with mean value of zero and standard deviation of  $\sigma_\alpha$ :

$$P(\alpha) = ce^{-\frac{\alpha^2}{2\sigma_\alpha^2}} \quad (26)$$

where  $c$  is a constant. The standard deviation  $\sigma_\alpha$  represents the roughness of surface – the larger  $\sigma_\alpha$ , the rougher surface, and vice versa.

Figure 6.25 shows the coordinate system used in the Torrance-Sparrow model. A surface patch  $dA$  is located at the origin of the coordinate system with its normal vector coinciding with the  $Z$  axis. The surface is illuminated by the incident beam that lies on the  $YZ$  plane with a polar angle of  $\theta_i$ , and a particular reflected beam which we are interested in travels along the direction  $(\theta_r, \phi_r)$ . Unit solid angles  $d\omega_i$  and  $d\omega_r$  are used to denote the directions of the incident beam and the reflected beam respectively. The bisector between the incident direction and the reflected direction is described by a unit solid angle  $d\omega'$  which has a polar angle of  $\alpha$ .

Only the micro-facets in  $dA$  with normal vectors within  $d\omega'$  can reflect the incident light specularly to the direction  $(\theta_r, \phi_r)$ . Let  $P(\alpha)d\omega'$  be the number of facets per unit surface area whose normal vectors are contained within  $d\omega'$  where  $P(\alpha)$  was defined in Eq. (26). Then, the number of facets in  $dA$  with normal vectors lying within  $d\omega'$  is

$$P(\alpha)d\omega'dA.$$

Let  $a_f$  be the area of each micro-facet. Then, the total reflecting area of the facets is

$$a_f P(\alpha) d\omega' dA,$$

and the projected area in the incident direction is

$$a_f P(\alpha) d\omega' dA \cos \theta'_i.$$

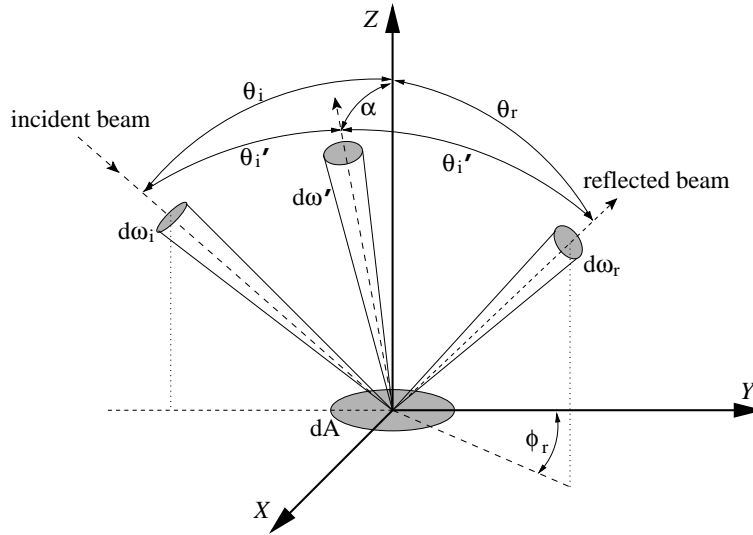


Fig. 6.25: Coordinate system for the Torrance-Sparrow model

Thus, the incident radiance of the specularly reflecting facets in  $dA$  is

$$L_i = \frac{d^2\Phi_i}{d\omega_i(a_f P(\alpha)d\omega' dA) \cos \theta_i'} \quad (27)$$

where  $\Phi_i$  is the incident flux. Since a surface is not a perfect reflector, only a fraction of the incident flux is reflected. Therefore, Torrance and Sparrow considered two phenomena for relating the incident flux and the reflected flux. First, they considered *Fresnel reflection coefficient*  $F'(\theta_i', \eta')$  [3], which determines the fraction of incident light that is reflected by a surface.  $\theta_i'$  represents the incident angle and  $\eta'$  represents the complex index of refraction of the surface. The Fresnel reflection coefficient is sufficient for relating the incident and reflected flux when facet shadowing and masking (See Figure 6.26) are neglected. For the second phenomenon, Torrance and Sparrow considered the effects of facet shadowing and masking, and introduced the *geometrical attenuation factor*  $G(\theta_i, \theta_r, \phi_r)$ <sup>3</sup>. On the basis of these two phenomena, the incident flux  $\Phi_i$  and the reflected flux  $\Phi_r$  can be related as

$$d^2\Phi_r = F'(\theta_i', \eta')G(\theta_i, \theta_r, \phi_r)d^2\Phi_i. \quad (28)$$

<sup>3</sup>Readers are referred to Torrance and Sparrow's paper [54] for the detailed description of geometrical attenuation factor.

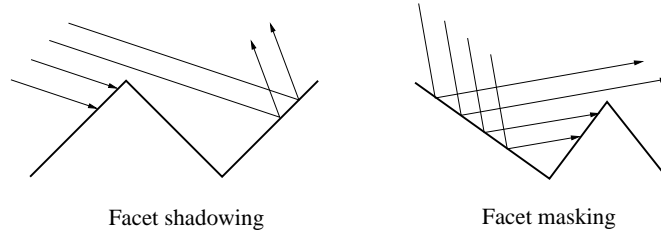


Fig. 6.26: Facet shadowing and masking

Since the radiance reflected in the direction  $(\theta_r, \phi_r)$  is given by

$$L_r = \frac{d^2\Phi_r}{d\omega_r dA \cos \theta_r},$$

using Eq. (27) and Eq. (28), the above equation can be rewritten as

$$L_r = \frac{F'(\theta'_i, \eta') G(\theta_i, \theta_r, \phi_r) L_i d\omega_i (c_f P(\alpha) d\omega' dA) \cos \theta'_i}{d\omega_r dA \cos \theta_r}. \quad (29)$$

The solid angles  $d\omega_r$  and  $d\omega'$  are related as

$$d\omega' = \frac{d\omega_r}{4 \cos \theta'_i},$$

thus by rewriting Eq. (29), we have

$$L_r = K_{spec} \frac{L_i d\omega_i}{\cos \theta_r} e^{-\frac{\alpha^2}{2\sigma^2}}, \quad (30)$$

where

$$K_{spec} = \frac{c_f F'(\theta'_i, \eta') G(\theta_i, \theta_r, \phi_r)}{4}$$

In order to account for the diffusely reflecting light, Torrance and Sparrow added the Lambertian model to Eq. (30):

$$L_r = K_{diff} L_i d\omega_i \cos \theta_i + K_{spec} \frac{L_i d\omega_i}{\cos \theta_r} e^{-\frac{\alpha^2}{2\sigma^2}}, \quad (31)$$

This equation describes the general Torrance-Sparrow reflection model.

### Nayar's Unified Model

By comparing the Beckmann-Spizzichino model and the Torrance-Sparrow model, a unified reflectance framework that is suitable for machine vision applications was developed [40]. In particular, this model consists of three components: diffuse lobe; specular lobe; and specular spike. The diffuse lobe represents the internal scattering mechanism, and is distributed around the surface normal. The specular lobe represents the reflection of incident light, and is distributed around the specular direction. Finally, the specular spike represents mirror-like reflection on smooth surfaces, and is concentrated along the specular direction. In machine vision, we are interested in image irradiance (intensity) values. Assuming that the object distance is much larger than both the focal length and the diameter of lens of imaging sensor (e.g., CCD camera), then it is shown that image irradiance is proportional to surface radiance. Therefore, the image intensity is given as a linear combination of the three reflection components:

$$I = I_{dl} + I_{sl} + I_{ss} \quad (32)$$

Two specific reflectance models were developed – one for the case of fixed light source with moving sensor and the other for the case of moving light source with fixed sensor. Figure 6.27 illustrates the reflectance model for the case of fixed light source and moving sensor. In this case, the image intensity observed by the sensor is given by

$$I = C_{dl} + C_{sl} \frac{1}{\cos \theta_r} e^{-\frac{\alpha^2}{2\sigma_\alpha^2}} + C_{ss} \delta(\theta_i - \theta_r) \delta(\phi_r) \quad (33)$$

where the constants  $C_{dl}$ ,  $C_{sl}$  and  $C_{ss}$  represent the strengths of the diffuse lobe, specular lobe and specular spike respectively, and  $\delta$  is a delta function. Pictorially, the strength of each reflection component is the magnitude of intersection point between the component contour and the viewing ray from the sensor. Notice that the strength of diffuse lobe is the same for all directions. Notice also that the peak of specular lobe is located at the angle slightly greater than the specular direction. This phenomenon is called *off-specular peak*, and it is caused by  $\frac{1}{\cos \theta_r}$  in the specular lobe component term in Eq. (33). The off angle between the specular direction and the peak direction of specular lobe becomes larger for rougher surfaces.

Figure 6.28 illustrates the reflectance model for the case of moving source light and fixed sensor, and the image intensity observed by the sensor in this case is given by

$$I = K_{dl} \cos \theta_i + K_{sl} e^{-\frac{\alpha^2}{2\sigma_\alpha^2}} + K_{ss} \delta(\theta_i - \theta_r) \delta(\phi_r) \quad (34)$$

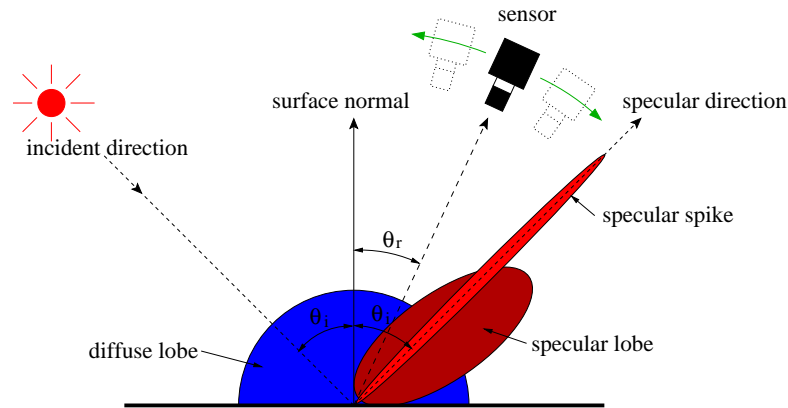


Fig. 6.27: Reflectance model for the case of fixed light source and moving sensor

It is important to note that the pictorial illustration of the strength of diffuse lobe component is different from the previous case whereas the strengths of specular lobe and specular spike are the same. Specifically, the strength of the diffuse lobe component is the magnitude of the intersection point between the diffuse lobe contour and the incident light ray, not the viewing ray as in the previous case. Notice that  $\theta_r$  is constant since the sensor is fixed. Therefore,  $\frac{1}{\cos \theta_r}$  can be added to the constant term of the specular lobe component (i.e.,  $K_{sl}$ ). Consequently, the off-specular peak is no longer observed in this case. Eq. (34) is useful for acquiring reflectance property of an object using the photometric stereo method.

The specular lobe constants  $C_{sl}$  in Eq. (33) and  $K_{sl}$  in Eq. (34) represent  $K_{spec}$  in Eq. (31). Clearly,  $K_{spec}$  is not a constant since it is a function of the Fresnel reflection coefficient  $F'(\theta_i', \eta')$  and the geometrical attenuation factor  $G(\theta_i, \theta_r, \phi_r)$ . However, the Fresnel reflection coefficient is nearly constant until  $\theta_i'$  becomes  $90^\circ$ , and the geometrical attenuation factor is 1 as long as both  $\theta_i$  and  $\theta_r$  are within  $45^\circ$ . Thus, assuming that  $\theta_i'$  is less than  $90^\circ$  and  $\theta_i$  and  $\theta_r$  are less than  $45^\circ$ ,  $C_{sl}$  and  $K_{sl}$  can be considered to be constants.

### **Ambient-Diffuse-Specular Model**

The ambient-diffuse-specular model, despite its inaccuracy in representing reflectance properties of real object surfaces, is currently the most commonly used reflectance model in the computer graphics community. The main attraction of this model is its simplicity. It describes the reflected light on

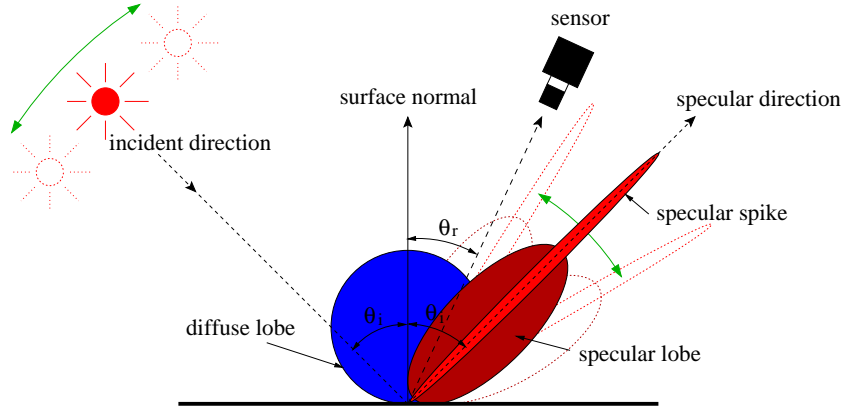


Fig. 6.28: Reflectance model for the case of moving light source and fixed sensor

the object point as a mixture of ambient, diffuse (or body), and specular (or surface) reflection. Roughly speaking, the ambient reflection represents the global reflection property that is constant for entire scene, the diffuse reflection represents the property that plays the most important role in determining what is perceived as the “true” color, and the specular reflection represents bright spots, or highlights caused by the light source. Most commonly used computer graphics applications (e.g., OpenGL) formulate the ambient-diffuse-specular model as

$$I = I_a K_a + I_l K_d \cos \theta + I_l K_s \cos^n \alpha \quad (35)$$

where  $I_a$  and  $I_l$  are the intensities of ambient light and light source respectively, and  $K_a$ ,  $K_d$  and  $K_s$  are constants that represent the strengths of ambient, diffuse and specular components respectively.  $\theta$  is the angle between the light source direction and the surface normal direction of the object point,  $\alpha$  is the angle between the surface normal and the bisector of the light source and the viewing direction, and  $n$  is a constant that represents the “shininess” of the surface.

Let  $\mathbf{L}$  be the light source direction,  $\mathbf{N}$  the surface normal,  $\mathbf{E}$  the viewing direction, and  $\mathbf{H}$  the bisector of  $\mathbf{L}$  and  $\mathbf{E}$  (see Figure 6.29), then assuming all vectors are unit vectors, we can rewrite Eq. (35) as

$$I = I_a K_a + I_l K_d (\mathbf{L} \cdot \mathbf{N}) + I_l K_s (\mathbf{H} \cdot \mathbf{N})^n \quad (36)$$

where  $\cdot$  is a dot product.

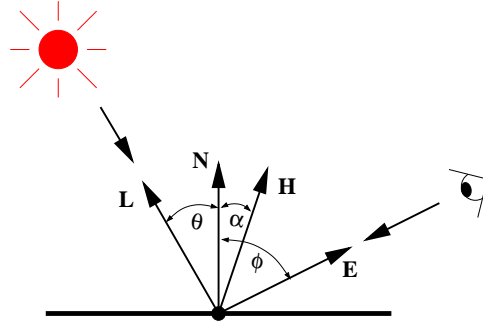


Fig. 6.29: Basic light reflection model.

### 6.5.2 Reflection Model Parameter Estimation

Ikeuchi and Sato [27] presented a system for determining reflectance properties of an object using a single pair of range and intensity images. The range and intensity images are acquired by the same sensor, thus the correspondence between the two images are directly provided. That is, 3D position, normal direction, and intensity value for each data point are available. The reflectance model they used is similar to that of Nayar's unified model [40], but only considered the diffuse lobe and the specular lobe:

$$I = K_d \cos \theta_i + K_s \frac{1}{\cos \theta_r} e^{-\frac{\alpha^2}{2\sigma_\alpha^2}} \quad (37)$$

Assuming the object's reflectance property is uniform over the surface, their system estimates four variables: light source direction  $\mathbf{L} = [L_x \ L_y \ L_z]^T$ , diffuse component constant  $K_d$ , specular component constant  $K_s$  and surface roughness  $\sigma_\alpha$ . Let  $I(i, j)$  be the intensity value at  $i$ th row and  $j$ th column of the intensity image. The corresponding data point's normal is denoted as  $\mathbf{N}(i, j) = [N_x(i, j) \ N_y(i, j) \ N_z(i, j)]^T$ . Assuming the intensity image has no specular components, we have

$$\begin{aligned} I(i, j) &= K_d (\mathbf{L} \cdot \mathbf{N}(i, j)) \\ &= a N_x(i, j) + b N_y(i, j) + c N_z(i, j) \\ &= \mathbf{A} \cdot \mathbf{N}(i, j) \end{aligned}$$

where  $a = K_d L_x$ ,  $b = K_d L_y$ ,  $c = K_d L_z$  and  $\mathbf{A} = [a \ b \ c]^T$ . Then,  $\mathbf{A}$  is initially estimated using a least square fitting by minimizing the following

equation:

$$e_1 = \sum_{i,j} [I(i,j) - aN_x(i,j) - bN_y(i,j) - cN_z(i,j)]^2.$$

The estimated vector  $\mathbf{A}^* = [a^* \ b^* \ c^*]^T$  is used to determine the ideal diffuse brightness  $I'$  for each data point:

$$I'(i,j) = a^*N_x(i,j) + b^*N_y(i,j) + c^*N_z(i,j).$$

Based on the computed ideal diffuse brightness values, the pixels are categorized into three groups using a threshold: if the observed intensity is much greater than the ideal diffuse intensity, it is considered to be a *highlight pixel*; if the observed intensity is much less than the ideal diffuse intensity, it is considered to be a *shadow pixel*; and all other pixels are categorized as *diffuse pixels*. Using only the diffuse pixels, the vector  $\mathbf{A}$  and the ideal diffuse intensity values  $I'(i,j)$  are recomputed, and the process is repeated until  $\mathbf{A}^*$  converges. At the end, the diffuse component constant  $K_d$  and the direction of light source  $\mathbf{L}$  are given by

$$\begin{aligned} K_d &= \sqrt{a^{*2} + b^{*2} + c^{*2}} \\ \mathbf{L} &= \begin{bmatrix} a^* & b^* & c^* \\ K_d & K_d & K_d \end{bmatrix}^T \end{aligned}$$

The next step consists of estimating the specular parameters  $K_s$  and  $\sigma_\alpha$ . In order to estimate the specular parameters, the highlight pixels determined in the previous process are additionally divided into two subgroups, *specular* and *interreflection* pixels, based on the angle  $\alpha$  (Recall that  $\alpha$  is the angle between surface normal and the bisector of source light and viewing direction). If  $\alpha(i,j)$  is less than a threshold, the pixel is categorized as a specular pixel, and otherwise, it is considered to be a interreflection pixel. Intuitively, this criterion is due to the fact that mirror-like or close to mirror-like reflecting data points must have small  $\alpha$  values. If a point contains high intensity value with relatively large  $\alpha$ , we may assume that the main cause of the high intensity is not from the source light, but from interreflected lights. Let  $d(i,j)$  be a portion of intensity  $I(i,j)$  contributed by the specular component, and it is given by

$$\begin{aligned} d(i,j) &= K_s \frac{1}{\cos \theta_r(i,j)} e^{-\frac{\alpha^2(i,j)}{2\sigma_\alpha^2}} \\ &= I(i,j) - \mathbf{A} \cdot \mathbf{N}(i,j) \end{aligned}$$

The specular parameters  $K_s$  and  $\sigma_\alpha$  are estimated by employing a two-step

fitting method. The first step assumes that  $K_s$  is known, and estimates  $\sigma_\alpha$  by minimizing

$$e_2 = \sum_{i,j} \left[ \ln d'(i,j) - \ln K_s + \ln(\cos \theta_r(i,j)) + \frac{\alpha^2(i,j)}{2\sigma_\alpha^2} \right]^2$$

where  $d'(i,j) = I(i,j) - \mathbf{A}^* \cdot \mathbf{N}(i,j)$ . Given  $\sigma_\alpha$ , the second step estimates  $K_s$  by minimizing

$$e_3 = \sum_{i,j} \left[ d'(i,j) - K_s \frac{1}{\cos \theta_r(i,j)} e^{-\frac{\alpha^2(i,j)}{2\sigma_\alpha^2}} \right]^2.$$

By repeating the two steps, the specular parameters  $K_s$  and  $\sigma_\alpha$  are estimated.

Sato and Ikeuchi [48] extended the above system for multiple color images of a geometric model generated from multiple range images. They first acquire a small number of range images by rotating the object on a rotary stage, and generate a 3D model. Then, they acquire color images of the object, but this time they acquire more color images than the range images by rotating the object with a smaller interval between images.<sup>4</sup> The correspondence between the 3D model and the color images are known since the same sensor is used for both range and color image acquisition, and since each image was taken at a known rotation angle without moving the object. Specifically, the 3D model can be projected onto a color image using the 4 by 3 camera projection matrix rotated by the angle in which the image was taken. The light source is located near the sensor, thus they assume that the light source direction is the same as the viewing direction. Consequently, the angles  $\theta_r$ ,  $\theta_i$  and  $\alpha$  are all the same, and the reflectance model is given by

$$I = K_d \cos \theta + K_s \frac{1}{\cos \theta} e^{-\frac{\theta^2}{2\sigma_\alpha^2}} \quad (38)$$

In order to estimate the reflectance parameters, they first separate the diffuse components from the specular components. Let  $\mathbf{M}$  be a series of intensity values of a data point observed from  $n$  different color images:

$$\mathbf{M} = \begin{bmatrix} I_1 \\ I_2 \\ \vdots \\ I_n \end{bmatrix} = \begin{bmatrix} I_{1,R} & I_{1,G} & I_{1,B} \\ I_{2,R} & I_{2,G} & I_{2,B} \\ \vdots & \vdots & \vdots \\ I_{n,R} & I_{n,G} & I_{n,B} \end{bmatrix}$$

<sup>4</sup>8 range images (45° interval) and 120 color images (3° interval) were acquired in the example presented in their paper

where the subscripts  $R$ ,  $G$  and  $B$  represent three primary colors. Using Eq. (38),  $\mathbf{M}$  can be expressed as

$$\begin{aligned} \mathbf{M} &= \begin{bmatrix} \cos \theta_1 & E(\theta_1) \\ \cos \theta_2 & E(\theta_2) \\ \vdots & \vdots \\ \cos \theta_n & E(\theta_n) \end{bmatrix} \begin{bmatrix} K_{d,R} & K_{d,G} & K_{d,B} \\ K_{s,R} & K_{s,G} & K_{s,B} \end{bmatrix} \\ &= \begin{bmatrix} \mathbf{G}_d & \mathbf{G}_s \end{bmatrix} \begin{bmatrix} \mathbf{K}_d^T \\ \mathbf{K}_s^T \end{bmatrix} \\ &= \mathbf{GK} \end{aligned}$$

where  $E(\theta_i) = \frac{1}{\cos \theta_i} e^{-\frac{\theta_i^2}{2\sigma_\alpha^2}}$ . By assuming  $\mathbf{K}_s$  is pure white (i.e.  $\mathbf{K}_s = [1 \ 1 \ 1]^T$ ) and  $\mathbf{K}_d$  is the color value with the largest  $\theta$  (i.e.,  $\mathbf{K}_d = [I_{i,R} \ I_{i,G} \ I_{i,B}]^T$ ) where  $\theta_i = \max(\theta_1, \theta_2, \dots, \theta_n)$ ,  $\mathbf{G}$  can be computed by

$$\mathbf{G} = \mathbf{MK}^+$$

where  $\mathbf{K}^+$  is a  $3 \times 2$  pseudo-inverse of  $\mathbf{K}$ . With the computed  $\mathbf{G}$ , we can separate the diffuse components  $\mathbf{M}_d$  and the specular components  $\mathbf{M}_s$  by

$$\begin{aligned} \mathbf{M}_d &= \mathbf{G}_d \mathbf{K}_d^T \\ \mathbf{M}_s &= \mathbf{G}_s \mathbf{K}_s^T. \end{aligned}$$

Then, the diffuse reflectance parameter  $\mathbf{K}_d$  and the specular reflectance parameters  $\mathbf{K}_s$  and  $\sigma_\alpha$  can be estimated by applying two separate fitting processes on  $\mathbf{M}_d$  and  $\mathbf{M}_s$ . However, the authors pointed out that the diffuse reflectance parameter was reliably estimated for each data point while the estimation of specular reflectance parameters, on the other hand, was unreliable because the specular component is usually observed from a limited range of viewing directions, and even if the specular component is observed, the parameter estimation can become unreliable if it is not observed strongly. Therefore, the specular reflectance parameters are estimated for each segmented region based on the hue value assuming that all the data points in each region are characterized by common specular reflectance parameters.<sup>5</sup>

In Sato *et al.* [49], instead of estimating common specular reflectance parameters for each segmented region, the authors simply select data points where specular component is observed sufficiently, and estimate parameters only on those points. The estimated parameters are then linearly interpolated over the entire object surface.

<sup>5</sup>The specular reflectance parameters were estimated in 4 different regions in the example in the paper.

Kay and Caelli [30] follow the idea of photometric stereo [61] and take multiple intensity images of a simple object from a single viewpoint but each time with a different light source position. The acquired intensity images along with a range image acquired from the same viewpoint are used to estimate reflectance parameters for each data point. Since all the intensity images and the range image are acquired from the same viewpoint, the problem of registration is avoided. Like [27], they also categorize each data point by the amount of information needed for the parameter estimation. If a data point contains sufficient information, the reflectance parameters are computed by fitting the data to the reflectance model similar to Eq (37), otherwise the parameters are interpolated.

Lensch *et al.* [33] first generate a 3D model of an object, and acquire several color images of the object from different viewpoints and light source positions. In order to register the 3D model and the color images, a silhouette based registration method described in their earlier paper [32] is used. Given the 3D model, and multiple radiance samples of each data point obtained from color images, reflectance parameters are estimated by fitting the data into the reflectance model proposed by Lafortune *et al.* [31]. For reliable estimation, the reflectance parameters are computed for each cluster of similar material. The clustering process initially begins by computing a set of reflectance parameters,  $\mathbf{a}$ , that best fits the entire data. The covariance matrix of the parameters obtained from the fitting, which provides the distribution of fitting error, is used to generate two new clusters. Specifically, two sets of reflectance parameters,  $\mathbf{a}_1$  and  $\mathbf{a}_2$ , are computed by shifting  $\mathbf{a}$  in the parameter space along the eigenvector corresponding to the largest eigenvalue of the covariance matrix. That is,

$$\mathbf{a}_1 = \mathbf{a} + \tau \mathbf{e} \quad \mathbf{a}_2 = \mathbf{a} - \tau \mathbf{e}$$

where  $\mathbf{e}$  is the largest eigenvector and  $\tau$  is a constant. The data are then redistributed into two clusters based on the magnitude of fitting residuals to  $\mathbf{a}_1$  and  $\mathbf{a}_2$ . However, due to the data noise and improper scaling of  $\tau$ , the split will not be optimal, and the two new clusters may not be clearly separated. Thus, the splitting process includes an iteration of redistributing the data based on  $\mathbf{a}_1$  and  $\mathbf{a}_2$ , and recomputing  $\mathbf{a}_1$  and  $\mathbf{a}_2$  by fitting the data of the corresponding cluster. The iteration terminates when the members of both clusters do not change any more. The splitting process is repeatedly performed on a new cluster until the number of clusters reaches a prespecified number. The clustering process results reflectance parameters for each cluster of similar material. Although applying a single set of reflectance parameters for each cluster would yield a plausible result, the authors provided a method for generating point by point variations within a cluster. The idea is to represent each point by a linear combination of the elements of the basis set of reflectance parameters. The basis set include the original reflectance

parameters computed for the cluster, the reflectance parameters of neighboring clusters, the reflectance parameters of similar clusters, and reflectance parameters generated by slightly increasing or decreasing the original values. The authors pointed out that the use of a linear basis set in most cases does not improve upon the results achieved with the original reflectance parameter set.

Levoy *et al.* [35], in their Digital Michelangelo Project, also employ two different passes for the acquisition of geometric data and color images. The registration problem between the color images and the geometric model is solved by maintaining the position and the orientation of the camera with respect to the range sensor at all times. Since the acquisition process had to be performed inside the museum, the lighting condition could not be controlled. This implies that the ambient light had to be considered as well. To get around this problem, they took two images from identical camera position, but one image only under the ambient light, and the other under the ambient light together with the calibrated light source. Then, subtracting the first image from the second results an image that represents what the camera would have seen only with the calibrated light source. After acquiring color images covering the entire surface, the systematic camera distortions of the images such as geometric distortion and chromatic aberration are corrected. Next, pixels that were occluded with respect to the camera or the light source are discarded. Finally, the remaining pixels are projected onto the merged geometric data for estimating the reflection parameters. They followed an approach similar to that described in [49], except that they only extracted diffuse reflection parameters. To eliminate specular contributions, they additionally discarded pixels that were observed with small  $\alpha$  (i.e., close to mirror reflection direction).

## 6.6 Conclusion

In this report, we have presented the state-of-the-art methods for constructing geometrically and photometrically correct 3D models of real-world objects using range and intensity images. We have described four general steps involved in 3D modeling where each respective step continues to be an active research area on its own in the computer vision and computer graphics communities.

Although recent research efforts established the feasibility of constructing photo-realistic 3D models of physical objects, the current techniques are capable of modeling only a limited range of objects. One source of this limitation is severe self-occlusions, which make certain areas of object very difficult to be reached by the sensors. Another source of difficulty is the fact that many real-world objects have complex surface materials that cause problems particularly in range data acquisition and in reflectance property

estimation. Various surface properties that cause difficulties in range data acquisition include specular surfaces, highly absorptive surfaces, translucent surfaces and transparent surfaces. In order to ensure that the object surface is ideal for range imaging, some researchers have simply painted the object or coated the object with removable powder. Obviously, such approaches may not be desirable or even possible outside laboratories. Park and Kak [43] recently developed a new range imaging method that accounts for the effects of mutual reflections, thus providing a way to construct accurate 3D models even of specular objects.

Complex surface materials also cause problems in reflectance property estimation. As we mentioned earlier, a large number of samples is needed in order to make a reliable estimation of reflectance property, and acquiring sufficient samples for each point of the object is very difficult. Therefore, some methods assume the object to have uniform reflectance property while other methods estimate reflectance properties only for the points with sufficient samples and linearly interpolate the parameters throughout the entire surface. Or yet, some methods segment the object surface into groups of similar materials and estimate reflectance property for each group. As one can expect, complex surface materials with high spatial variations can cause unreliable estimation of reflectance property.

The demand for constructing 3D models of various objects has been steadily growing and we can naturally predict that it will continue to grow in the future. Considering all innovations in 3D modeling we have seen in recent years, we believe the time when machines take a random object and automatically generate its replica is not too far away.

## References

- [1] N. Amenta, M. Bern, and M. Kamvysselis. A new voronoi-based surface reconstruction algorithm. In *SIGGRAPH'98*, pages 415–412, 1998.
- [2] C. Bajaj, F. Bernardini, and G. Xu. Automatic reconstruction of surfaces and scalar fields from 3D scans. In *SIGGRAPH'95*, pages 109–118, 1995.
- [3] P. Beckmann and A. Spizzichino. *The Scattering of Electromagnetic Waves from Rough Surfaces*. Pergamon Press, 1963.
- [4] R. Benjemaa and F. Schmitt. Fast global registration of 3D sampled surfaces using a multi-Z-buffer technique. In *Conference on Recent Advances in 3-D Digital Imaging and Modeling*, pages 113–120, 1997.

- [5] R. Bergevin, M. Soucy, H Gagnon, and D. Laurendeau. Towards a general multiview registration technique. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(5):540–547, 1996.
- [6] M. Bern and D. Eppstein. Mesh generation and optimal triangulation. Technical Report P92-00047, Xerox Palo Alto Research Center, 1992.
- [7] F. Bernardini, J. Mittleman, H. Rushmeier, C. Silva, and G. Taubin. The ball-pivoting algorithm for surface reconstruction. *IEEE Transactions on Visualization and Computer Graphics*, 5(4):349–359, 1999.
- [8] P. J. Besl and N. D. McKay. A method for registration of 3-D shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(2):239–256, 1992.
- [9] F. Blais and M. Rioux. Real-time numerical peak detector. *Signal Processing*, 11:145–155, 1986.
- [10] J-D Boissonnat. Geometric structure for three-dimensional shape representation. *ACM Transactions on Graphics*, 3(4):266–286, 1984.
- [11] C. Chen, Y. Hung, and J. Chung. A fast automatic method for registration of partially-overlapping range images. In *IEEE International Conference on Computer Vision*, pages 242–248, 1998.
- [12] Y. Chen and G. Medioni. Object modeling by registration of multiple range images. In *IEEE International Conference on Robotics and Automation*, pages 2724–2729, 1991.
- [13] Y. Chen and G. Medioni. Object modeling by registration of multiple range images. *Image and Vision Computing*, 14(2):145–155, 1992.
- [14] B. Curless and M. Levoy. A volumetric method for building complex models from range images. In *SIGGRAPH'96*, pages 303–312, 1996.
- [15] P. E. Debevec, C. J. Taylor, and J. Malik. Modeling and rendering architecture from photographs: A hybrid geometry- and image-based approach. In *SIGGRAPH'96*, pages 11–20, 1996.
- [16] H. Edelsbrunner and E. P. Mucke. Three-dimensional alpha shapes. *ACM Transactions on Graphics*, 13(1):43–72, 1994.
- [17] O. Faugeras and M. Hebert. The representation, recognition, and locating of 3D shapes from range data. *The International Journal of Robotics Research*, 5(3):27–52, 1986.
- [18] H. Gagnon, M. Soucy, R. Bergevin, and D. Laurendeau. Registration of multiple range views for automatic 3-D modeling building. In *IEEE Computer Vision and Pattern Recognition*, pages 581–586, 1994.

- [19] G. Godin and P. Boulanger. Range image registration through invariant computation of curvature. In *ISPRS Workshop: From Pixels to Sequences*, pages 170–175, 1995.
- [20] G. Godin, D. Laurendeau, and R. Bergevin. A method for the registration of attributed range images. In *Third International Conference on 3-D Digital Imaging and Modeling*, pages 179–186, 2001.
- [21] G. Godin, M. Rioux, and R. Baribeau. 3-D registration using range and intensity information. In *SPIE Videometrics III*, pages 279–290, 1994.
- [22] M. Hebert, K. Ikeuchi, and H. Delingette. A spherical representation for recognition of free-form surfaces. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(7):681–690, 1995.
- [23] A. Hilton, A. Stoddart, J. Illingworth, and T. Windeatt. Marching triangles: Range image fusion for complex object modeling. In *IEEE International Conference on Image Processing*, pages 381–384, 1996.
- [24] H. Hoppe, T. DeRose, T. Duchamp, J McDonald, and W. Stuelzle. Surface reconstruction from unorganized points. In *SIGGRAPH'92*, pages 71–78, 1992.
- [25] B. K. P. Horn. Closed-form solution of absolute orientation using unit quaternions. *Optical Society of America A*, 4(4):629–642, April 1987.
- [26] D. Huber and M. Hebert. Fully automatic registration of multiple 3d data sets. *Image and Vision Computing*, 21(7):637–650, July 2003.
- [27] K. Ikeuchi and K. Sato. Determining reflectance properties of an object using range and brightness images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(11):1139–1153, November 1991.
- [28] A. Johnson and M. Hebert. Surface registration by matching oriented points. In *Conference on Recent Advances in 3-D Digital Imaging and Modeling*, pages 121–128, 1997.
- [29] A. Johnson and S. Kang. Registration and integration of textured 3-D data. In *Conference on Recent Advances in 3-D Digital Imaging and Modeling*, pages 234–241, 1997.
- [30] G. Kay and T. Caelli. Inverting an illumination model from range and intensity maps. *CVGIP: Image Understanding*, 59(2):183–201, March 1994.
- [31] E. P. E. Lafortune, S.-C. Foo, K. E. Torrance, and D. P. Greenberg. Non-linear approximation of reflectance functions. In *SIGGRAPH'97*, pages 117–126, 1997.

- [32] H. P. A. Lensch, W. Heidrich, and H.-P. Seidel. Automated texture registration and stitching for real world models. In *The 8th Pacific Conference on Computer Graphics and Applications*, pages 317–326, 2000.
- [33] H. P. A. Lensch, J. Kautz, M. Goesele, W. Heidrich, and H.-P. Seidel. Image-based reconstruction of spatially varying materials. In *The 12th Eurographics Rendering Workshop*, 2001.
- [34] R. Lenz and R. Tsai. Techniques for calibration of the scale factor and image center for high frequency 3-D machine vision metrology. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 10(5):713–720, 1988.
- [35] M. Levoy, K. Pulli, B. Curless, Z. Rusinkiewicz, D. Koller, L. Pereira, M. Ginzton, S. Anderson, J. Davis, J. Ginsberg, J. Shade, and D. Fulk. The digital michelangelo project: 3D scanning of large statues. In *SIGGRAPH'00*, pages 131–144, 2000.
- [36] W. E. Lorensen and H. E. Cline. Marching cubes: A high resolution 3D surface construction algorithm. In *SIGGRAPH'87*, pages 163–169, 1987.
- [37] T. Masuda, K. Sakaue, and N. Yokoya. Registration and integration of multiple range images for 3-D model construction. In *IEEE International Conference on Pattern Recognition*, pages 879–883, 1996.
- [38] T. Masuda and N. Yokoya. A robust method for registration and segmentation of multiple range images. In *IEEE CAD-Based Vision Workshop*, pages 106–113, 1994.
- [39] C. Montani, R. Scateni, and R. Scopigno. A modified look-up table for implicit disambiguation of marching cubes. *Visual Computer*, 10(6):353–355, 1994.
- [40] S. K. Nayar, K. Ikeuchi, and T. Kanade. Surface reflection: Physical and geometrical perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(7):611–634, July 1991.
- [41] F. E. Nicodemus, J. C. Richmond, J. J. Hsia, I. W. Ginsberg, and T. Limperis. Geometrical considerations and nomenclature for reflectance. Technical Report BMS Monograph 160, National Bureau of Standards, October 1977.
- [42] K. Nishino, Y. Sato, and K. Ikeuchi. Eigen-texture method: Appearance compression based on 3D model. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, pages 618–624, 1999.

- [43] J. Park and A. C. Kak. Multi-peak range imaging for accurate 3D reconstruction of specular objects. In *6th Asian Conference on Computer Vision*, 2004.
- [44] M. Potmesil. Generating models of solid objects by matching 3D surface segments. In *The 8th International Joint Conference on Artificial Intelligence (IJCAI)*, pages 1089–1093, 1983.
- [45] K. Pulli. *Surface Reconstruction and Display from Range and Color Data*. PhD thesis, University of Washington, 1997.
- [46] K. Pulli. Multiview registration for large data sets. In *Second International Conference on 3-D Digital Imaging and Modeling*, pages 160–168, 1999.
- [47] K. Pulli, M. Cohen, T. Duchamp, H. Hoppe, L. Shapiro, and W. Stuetzle. View-based rendering: Visualizing real objects from scanned range and color data. In *8th Eurographics Workshop on Rendering*, pages 23–34, 1997.
- [48] Y. Sato and K. Ikeuchi. Reflectance analysis for 3D computer graphics model generation. *Graphical Models and Image Processing*, 58(5):437–451, September 1996.
- [49] Y. Sato, M. Wheeler, and K. Ikeuchi. Object shape and reflectance modeling from observation. In *SIGGRAPH'97*, pages 379–387, 1997.
- [50] T. Schuts, T. Jost, and H. Hugli. Multi-featured matching algorithm for free-form 3D surface registration. In *IEEE International Conference on Pattern Recognition*, pages 982–984, 1998.
- [51] M. Soucy and D. Laurendeau. Multi-resolution surface modeling from multiple range views. In *IEEE Computer Vision and Pattern Recognition*, pages 348–353, 1992.
- [52] M. Soucy and D. Laurendeau. A general surface approach to the integration of a set of range views. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(4):344–358, 1995.
- [53] A. Stoddart, S. Lemke, A. Hilton, and T. Penn. Estimating pose uncertainty for surface registration. In *British Machine Vision Conference*, pages 23–32, 1996.
- [54] K. E. Torrance and E. M. Sparrow. Theory for off-specular reflection from roughened surfaces. *Journal of the Optical Society of America A*, 57(9):1105–1114, September 1967.

- [55] E. Trucco, R. B. Fisher, A. W. Fitzgibbon, and D. K. Naidu. Calibration, data consistency and model acquisition with laser stripes. *International Journal of Computer Integrated Manufacturing*, 11(4):293–310, 1998.
- [56] R. Tsai. A versatile camera calibration technique for high-accuracy 3D machine vision metrology using off-the-shelf TV cameras and lenses. *IEEE Journal of Robotics and Automation*, 3(4):323–344, 1987.
- [57] G. Turk and M. Levoy. Zippered polygon meshes from range images. In *SIGGRAPH'94*, pages 311–318, 1994.
- [58] B. C. Vemuri and J. K. Aggarwal. 3D model construction from multiple views using range and intensity data. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 435–438, 1986.
- [59] M. Wheeler, Y. Sato, and K. Ikeuchi. Consensus surface for modeling 3D objects from multiple range images. In *IEEE International Conference on Computer Vision*, pages 917–924, 1998.
- [60] D. N. Wood, D. I. Azuma, K. Aldinger, B. Curless, T. Duchamp, D. H. Salesin, and W. Stuetzle. Surface light fields for 3D photography. In *SIGGRAPH'00*, pages 287–296, 2000.
- [61] R. J. Woodham. Photometric method for determining surface orientation from multiple images. *Optical Engineering*, 19(1):139–144, 1980.
- [62] Z. Zhang. Iterative point matching for registration of free-form curves and surfaces. *International Journal of Computer Vision*, 13(2):119–152, 1994.