Chapter 12

3D reconstruction

DOI 10.1007/978-1-84882-935-0_12, © Springer-Verlag London Limited 2011

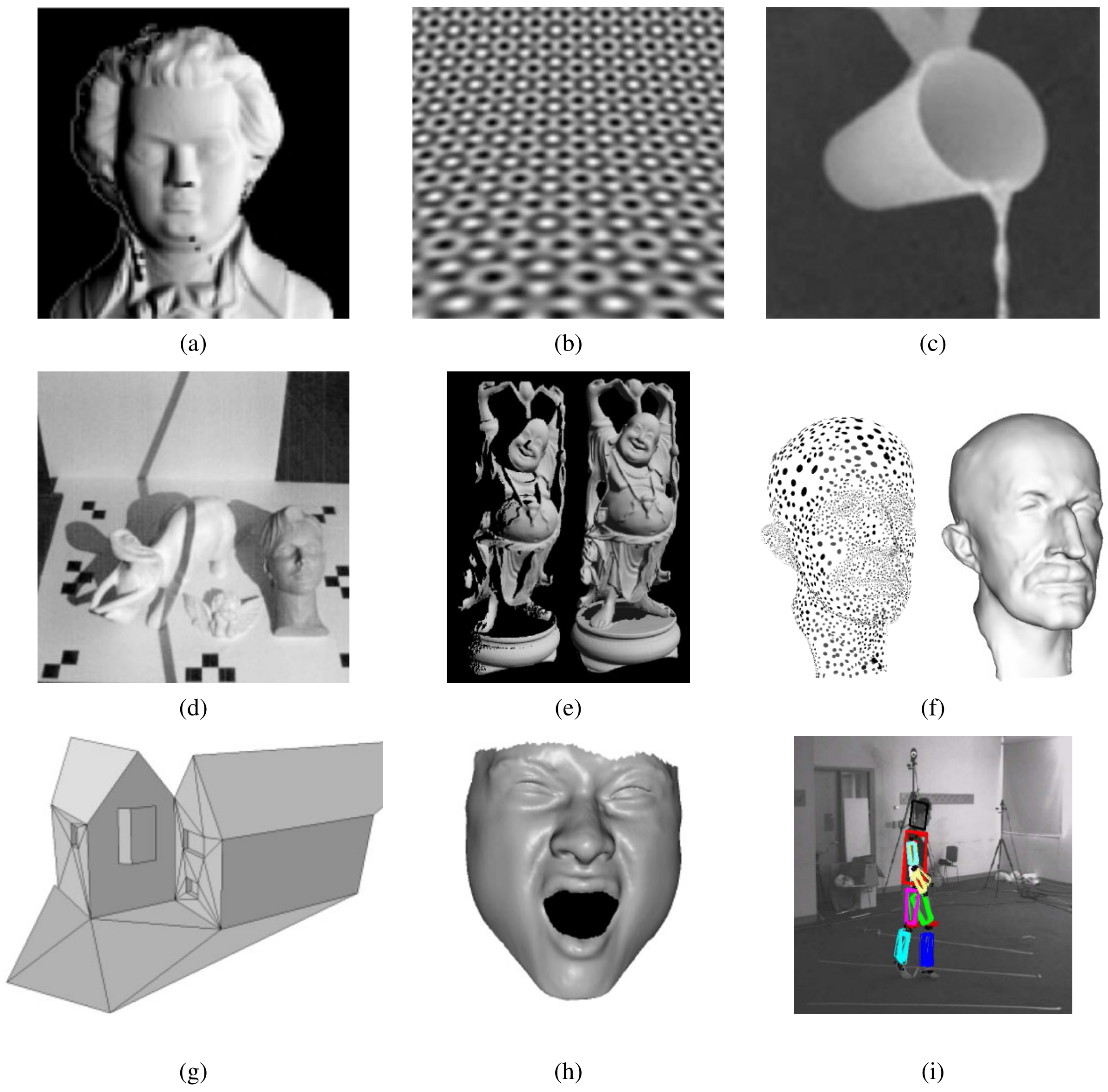


Figure 12.1 3D shape acquisition and modeling techniques: (a) shaded image (Zhang, Tsai, Cryer *et al.* 1999) © 1999 IEEE; (b) texture gradient (Garding 1992) © 1992 Springer; (c) real-time depth from focus (Nayar, Watanabe, and Noguchi 1996) © 1996 IEEE; (d) scanning a scene with a stick shadow (Bouguet and Perona 1999) © 1999 Springer; (e) merging range maps into a 3D model (Curless and Levoy 1996) © 1996 ACM; (f) point-based surface modeling (Pauly, Keiser, Kobbelt *et al.* 2003) © 2003 ACM; (g) automated modeling of a 3D building using lines and planes (Werner and Zisserman 2002) © 2002 Springer; (h) 3D face model from spacetime stereo (Zhang, Snavely, Curless *et al.* 2004) © 2004 ACM; (i) person tracking (Sigal, Bhatia, Roth *et al.* 2004) © 2004 IEEE.

As we saw in the previous chapter, a variety of stereo matching techniques have been developed to reconstruct high quality 3D models from two or more images. However, stereo is just one of the many potential cues that can be used to infer shape from images. In this chapter, we investigate a number of such techniques, which include not only visual cues such as shading and focus, but also techniques for merging multiple range or depth images into 3D models, as well as techniques for reconstructing specialized models, such as heads, bodies, or architecture.

Among the various cues that can be used to infer shape, the shading on a surface (Figure 12.1a) can provide a lot of information about local surface orientations and hence overall surface shape (Section 12.1.1). This approach becomes even more powerful when lights shining from different directions can be turned on and off separately (*photometric stereo*). Texture gradients (Figure 12.1b), i.e., the foreshortening of regular patterns as the surface slants or bends away from the camera, can provide similar cues on local surface orientation (Section 12.1.2). Focus is another powerful cue to scene depth, especially when two or more images with different focus settings are used (Section 12.1.3).

3D shape can also be estimated using active illumination techniques such as light stripes (Figure 12.1d) or time of flight range finders (Section 12.2). The partial surface models obtained using such techniques (or passive image-based stereo) can then be merged into more coherent 3D surface models (Figure 12.1e), as discussed in Section 12.2.1. Such techniques have been used to construct highly detailed and accurate models of cultural heritage such as historic sites (Section 12.2.2). The resulting surface models can then be simplified to support viewing at different resolutions and streaming across the Web (Section 12.3.2). An alternative to working with continuous surfaces is to represent 3D surfaces as dense collections of 3D oriented points (Section 12.4) or as volumetric primitives (Section 12.5).

3D modeling can be more efficient and effective if we know something about the objects we are trying to reconstruct. In Section 12.6, we look at three specialized but commonly occurring examples, namely architecture (Figure 12.1g), heads and faces (Figure 12.1h), and whole bodies (Figure 12.1i). In addition to modeling people, we also discuss techniques for tracking them.

The last stage of shape and appearance modeling is to extract some textures to paint onto our 3D models (Section 12.7). Some techniques go beyond this and actually estimate full BRDFs (Section 12.7.1).

Because there exists such a large variety of techniques to perform 3D modeling, this chapter does not go into detail on any one of these. Readers are encouraged to find more information in the cited references or more specialized publications and conferences devoted to these topics, e.g., the International Symposium on 3D Data Processing, Visualization, and Transmission (3DPVT), the International Conference on 3D Digital Imaging and Modeling (3DIM), the International Conference on Automatic Face and Gesture Recognition (FG), the IEEE Workshop on Analysis and Modeling of Faces and Gestures, and the International Workshop on Tracking Humans for the Evaluation of their Motion in Image Sequences (THEMIS).

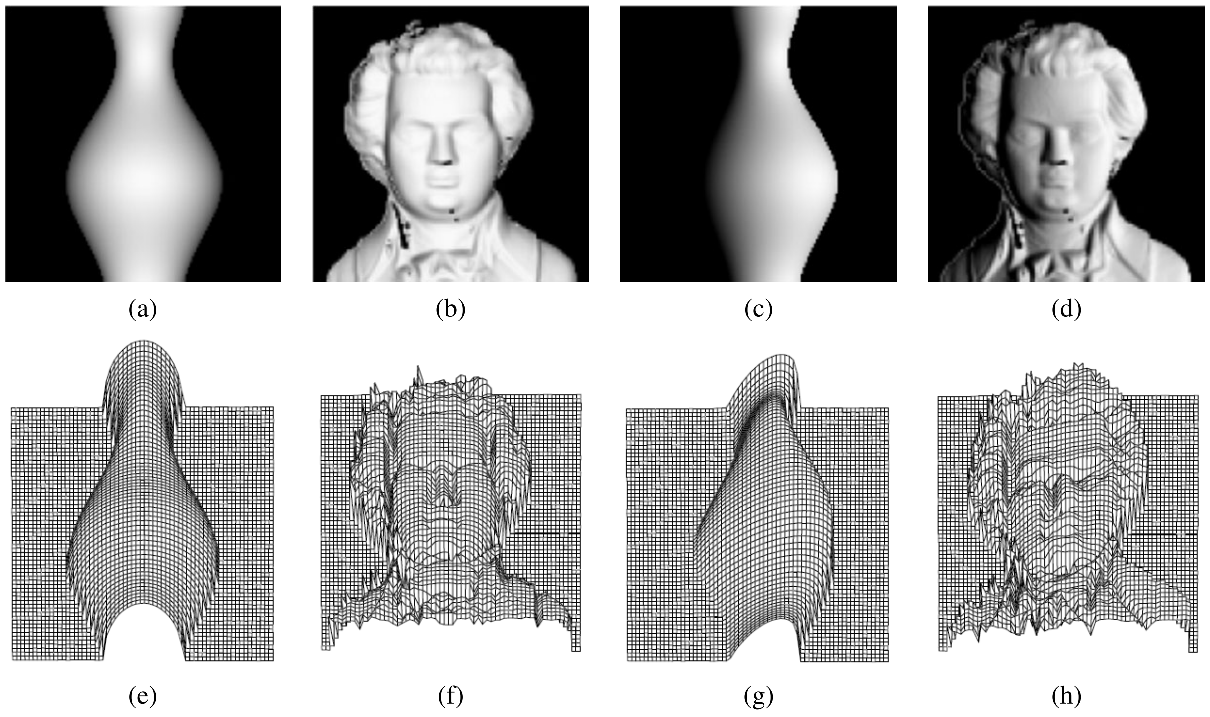


Figure 12.2 Synthetic shape from shading (Zhang, Tsai, Cryer *et al.* 1999) © 1999 IEEE: shaded images, (a–b) with light from in front $(0, 0, 1)$ and (c–d) with light the front right $(1, 0, 1)$; (e–f) corresponding shape from shading reconstructions using the technique of Tsai and Shah (1994).

12.1 Shape from X

In addition to binocular disparity, shading, texture, and focus all play a role in how we perceive shape. The study of how shape can be inferred from such cues is sometimes called *shape from X*, since the individual instances are called *shape from shading*, *shape from texture*, and *shape from focus*.[1] In this section, we look at these three cues and how they can be used to reconstruct 3D geometry. A good overview of all these topics can be found in the collection of papers on physics-based shape inference edited by Wolff, Shafer, and Healey (1992b).

12.1.1 Shape from shading and photometric stereo

When you look at images of smooth shaded objects, such as the ones shown in Figure 12.2, you can clearly see the shape of the object from just the shading variation. How is this possible? The answer is that as the surface normal changes across the object, the apparent brightness changes as a function of the angle between the local surface orientation and the incident illumination, as shown in Figure 2.15 (Section 2.2.2).

The problem of recovering the shape of a surface from this intensity variation is known as

[1] We have already seen examples of shape from stereo, shape from profiles, and shape from silhouettes in Chapter 11.

shape from shading and is one of the classic problems in computer vision (Horn 1975). The collection of papers edited by Horn and Brooks (1989) is a great source of information on this topic, especially the chapter on variational approaches. The survey by Zhang, Tsai, Cryer *et al.* (1999) not only reviews more recent techniques, but also provides some comparative results.

Most shape from shading algorithms assume that the surface under consideration is of a uniform albedo and reflectance, and that the light source directions are either known or can be calibrated by the use of a reference object. Under the assumptions of distant light sources and observer, the variation in intensity (*irradiance equation*) become purely a function of the local surface orientation,

$$I(x, y) = R(p(x, y), q(x, y)), \tag{12.1}$$

where $(p, q) = (z_x, z_y)$ are the depth map derivatives and $R(p, q)$ is called the *reflectance map*. For example, a diffuse (Lambertian) surface has a reflectance map that is the (non-negative) dot product (2.88) between the surface normal $\hat{\boldsymbol{n}} = (p, q, 1)/\sqrt{1 + p^2 + q^2}$ and the light source direction $\boldsymbol{v} = (v_x, v_y, v_z)$,

$$R(p, q) = \max\left(0, \rho \frac{pv_x + qv_y + v_z}{\sqrt{1 + p^2 + q^2}}\right), \tag{12.2}$$

where ρ is the surface reflectance factor (albedo).

In principle, Equations (12.1–12.2) can be used to estimate (p, q) using non-linear least squares or some other method. Unfortunately, unless additional constraints are imposed, there are more unknowns per pixel (p, q) than there are measurements (I). One commonly used constraint is the smoothness constraint,

$$\mathcal{E}_s = \int p_x^2 + p_y^2 + q_x^2 + q_y^2 \, dx \, dy = \int \|\nabla p\|^2 + \|\nabla q\|^2 \, dx \, dy, \tag{12.3}$$

which we already saw in Section 3.7.1 (3.94). The other is the *integrability constraint*,

$$\mathcal{E}_i = \int (p_y - q_x)^2 \, dx \, dy, \tag{12.4}$$

which arises naturally, since for a valid depth map $z(x, y)$ with $(p, q) = (z_x, z_y)$, we have $p_y = z_{xy} = z_{yx} = q_x$.

Instead of first recovering the orientation fields (p, q) and integrating them to obtain a surface, it is also possible to directly minimize the discrepancy in the image formation equation (12.1) while finding the optimal depth map $z(x, y)$ (Horn 1990). Unfortunately, shape from shading is susceptible to local minima in the search space and, like other variational problems that involve the simultaneous estimation of many variables, can also suffer from slow convergence. Using multi-resolution techniques (Szeliski 1991a) can help accelerate the convergence, while using more sophisticated optimization techniques (Dupuis and Oliensis 1994) can help avoid local minima.

In practice, surfaces other than plaster casts are rarely of a single uniform albedo. Shape from shading therefore needs to be combined with some other technique or extended in some way to make it useful. One way to do this is to combine it with stereo matching (Fua and Leclerc 1995) or known texture (surface patterns) (White and Forsyth 2006). The stereo and texture components provide information in textured regions, while shape from shading helps fill in the information across uniformly colored regions and also provides finer information about surface shape.

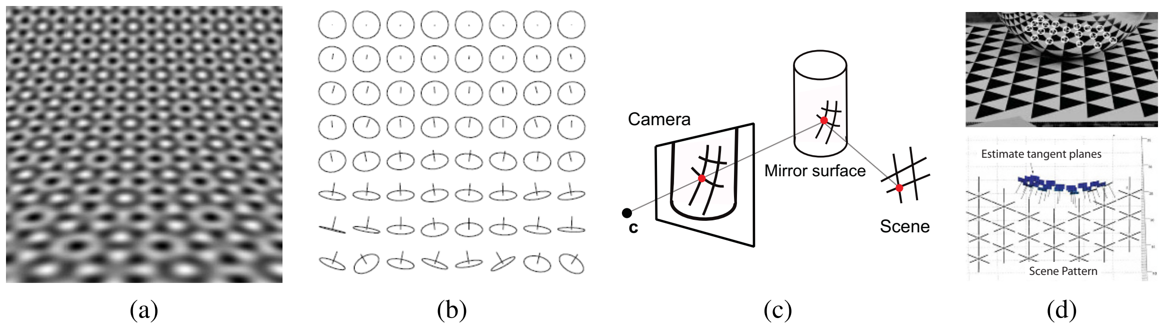


Figure 12.3 Synthetic shape from texture (Garding 1992) © 1992 Springer: (a) regular texture wrapped onto a curved surface and (b) the corresponding surface normal estimates. Shape from mirror reflections (Savarese, Chen, and Perona 2005) © 2005 Springer: (c) a regular pattern reflecting off a curved mirror gives rise to (d) curved lines, from which 3D point locations and normals can be inferred.

Photometric stereo. Another way to make shape from shading more reliable is to use multiple light sources that can be selectively turned on and off. This technique is called *photometric stereo*, since the light sources play a role analogous to the cameras located at different locations in traditional stereo (Woodham 1981).[2] For each light source, we have a different reflectance map, $R_1(p,q)$, $R_2(p,q)$, etc. Given the corresponding intensities I_1, I_2, etc. at a pixel, we can in principle recover both an unknown albedo ρ and a surface orientation estimate (p,q).

For diffuse surfaces (12.2), if we parameterize the local orientation by $\hat{\boldsymbol{n}}$, we get (for non-shadowed pixels) a set of linear equations of the form

$$I_k = \rho\hat{\boldsymbol{n}} \cdot \boldsymbol{v}_k, \tag{12.5}$$

from which we can recover $\rho\hat{\boldsymbol{n}}$ using linear least squares. These equations are well conditioned as long as the (three or more) vectors $\boldsymbol{v}_k$ are linearly independent, i.e., they are not along the same azimuth (direction away from the viewer).

Once the surface normals or gradients have been recovered at each pixel, they can be integrated into a depth map using a variant of regularized surface fitting (3.100). (Nehab, Rusinkiewicz, Davis *et al.* (2005) and Harker and O'Leary (2008) have produced some recent work in this area.)

When surfaces are specular, more than three light directions may be required. In fact, the irradiance equation given in (12.1) not only requires that the light sources and camera be distant from the surface, it also neglects inter-reflections, which can be a significant source of the shading observed on object surfaces, e.g., the darkening seen inside concave structures such as grooves and crevasses (Nayar, Ikeuchi, and Kanade 1991).

12.1.2 Shape from texture

The variation in foreshortening observed in regular textures can also provide useful information about local surface orientation. Figure 12.3 shows an example of such a pattern, along

[2] An alternative to turning lights on-and-off is to use three colored lights (Woodham 1994; Hernandez, Vogiatzis, Brostow *et al.* 2007; Hernandez and Vogiatzis 2010).

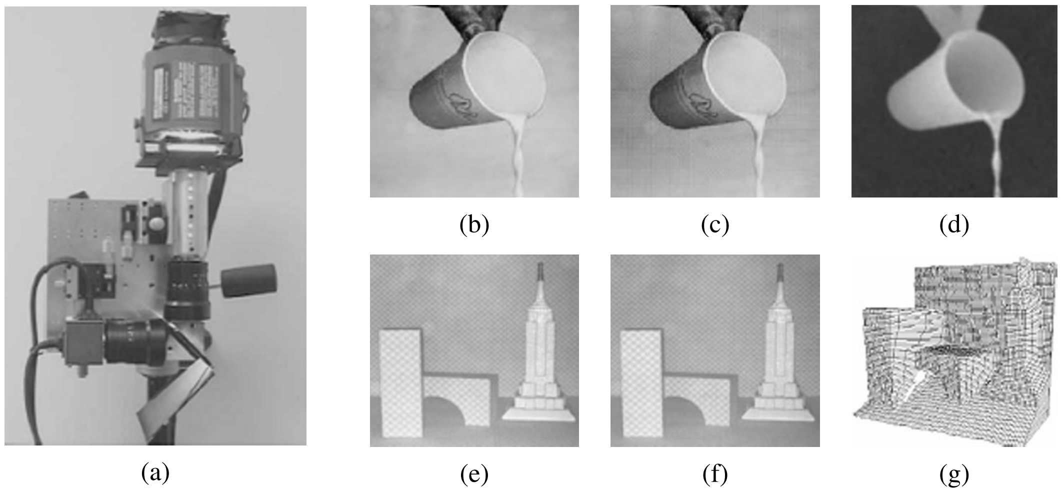


Figure 12.4 Real time depth from defocus (Nayar, Watanabe, and Noguchi 1996) © 1996 IEEE: (a) the real-time focus range sensor, which includes a half-silvered mirror between the two telecentric lenses (lower right), a prism that splits the image into two CCD sensors (lower left), and an edged checkerboard pattern illuminated by a Xenon lamp (top); (b–c) input video frames from the two cameras along with (d) the corresponding depth map; (e–f) two frames (you can see the texture if you zoom in) and (g) the corresponding 3D mesh model.

with the estimated local surface orientations. Shape from texture algorithms require a number of processing steps, including the extraction of repeated patterns or the measurement of local frequencies in order to compute local affine deformations, and a subsequent stage to infer local surface orientation. Details on these various stages can be found in the research literature (Witkin 1981; Ikeuchi 1981; Blostein and Ahuja 1987; Garding 1992; Malik and Rosenholtz 1997; Lobay and Forsyth 2006).

When the original pattern is regular, it is possible to fit a regular but slightly deformed grid to the image and use this grid for a variety of image replacement or analysis tasks (Liu, Collins, and Tsin 2004; Liu, Lin, and Hays 2004; Hays, Leordeanu, Efros *et al.* 2006; Lin, Hays, Wu *et al.* 2006; Park, Brocklehurst, Collins *et al.* 2009). This process becomes even easier if specially printed textured cloth patterns are used (White and Forsyth 2006; White, Crane, and Forsyth 2007).

The deformations induced in a regular pattern when it is viewed in the reflection of a curved mirror, as shown in Figure 12.3c–d, can be used to recover the shape of the surface (Savarese, Chen, and Perona 2005; Rozenfeld, Shimshoni, and Lindenbaum 2007). It is is also possible to infer local shape information from *specular flow*, i.e., the motion of specularities when viewed from a moving camera (Oren and Nayar 1997; Zisserman, Giblin, and Blake 1989; Swaminathan, Kang, Szeliski *et al.* 2002).

12.1.3 Shape from focus

A strong cue for object depth is the amount of blur, which increases as the object's surface moves away from the camera's focusing distance. As shown in Figure 2.19, moving the object surface away from the focus plane increases the circle of confusion, according to a formula that is easy to establish using similar triangles (Exercise 2.4).

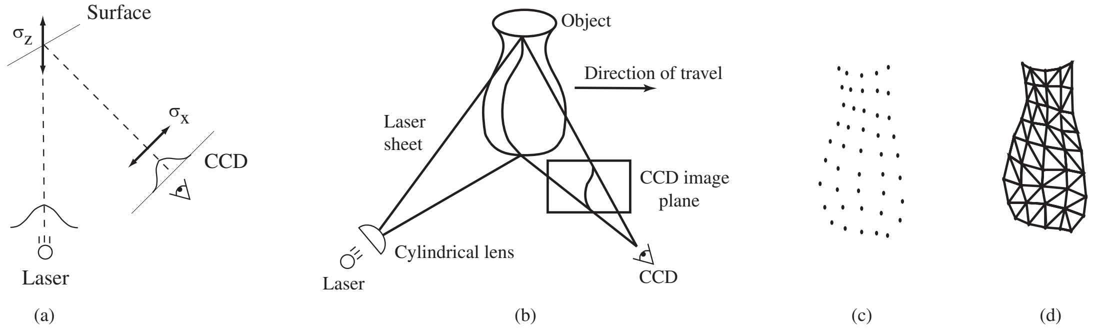


Figure 12.5 Range data scanning (Curless and Levoy 1996) © 1996 ACM: (a) a laser dot on a surface is imaged by a CCD sensor; (b) a laser stripe (sheet) is imaged by the sensor (the deformation of the stripe encodes the distance to the object); (c) the resulting set of 3D points are turned into (d) a triangulated mesh.

A number of techniques have been developed to estimate depth from the amount of defocus (*depth from defocus*) (Pentland 1987; Nayar and Nakagawa 1994; Nayar, Watanabe, and Noguchi 1996; Watanabe and Nayar 1998; Chaudhuri and Rajagopalan 1999; Favaro and Soatto 2006). In order to make such a technique practical, a number issues need to be addressed:

- The amount of blur increase in *both* directions as you move away from the focus plane. Therefore, it is necessary to use two or more images captured with different focus distance settings (Pentland 1987; Nayar, Watanabe, and Noguchi 1996) or to translate the object in depth and look for the point of maximum sharpness (Nayar and Nakagawa 1994).

- The magnification of the object can vary as the focus distance is changed or the object is moved. This can be modeled either explicitly (making correspondence more difficult) or using *telecentric optics*, which approximate an orthographic camera and require an aperture in front of the lens (Nayar, Watanabe, and Noguchi 1996).

- The amount of defocus must be reliably estimated. A simple approach is to average the squared gradient in a region but this suffers from several problems, including the image magnification problem mentioned above. A better solution is to use carefully designed *rational filters* (Watanabe and Nayar 1998).

Figure 12.4 shows an example of a real-time depth from defocus sensor, which employs two imaging chips at slightly different depths sharing a common optical path, as well as an active illumination system that projects a checkerboard pattern from the same direction. As you can see in Figure 12.4b–g, the system produces high-accuracy real-time depth maps for both static and dynamic scenes.

12.2 Active rangefinding

As we have seen in the previous section, actively lighting a scene, whether for the purpose of estimating normals using photometric stereo or for adding artificial texture for shape from

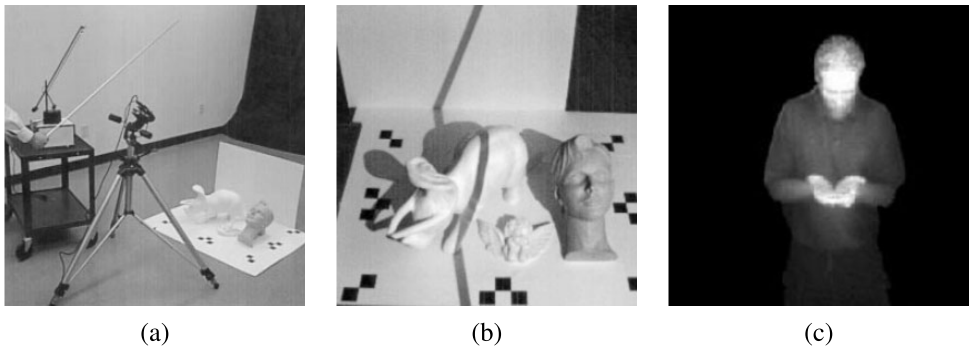


Figure 12.6 Shape scanning using cast shadows (Bouguet and Perona 1999) © 1999 Springer: (a) camera setup with a point light source (a desk lamp without its reflector), a hand-held stick casting a shadow, and (b) the objects being scanned in front of two planar backgrounds. (c) Real-time depth map using a pulsed illumination system (Iddan and Yahav 2001) © 2001 SPIE.

defocus, can greatly improve the performance of vision systems. This kind of *active illumination* has been used from the earliest days of machine vision to construct highly reliable sensors for estimating 3D depth images using a variety of *rangefinding* (or *range sensing*) techniques (Besl 1989; Curless 1999; Hebert 2000).

One of the most popular active illumination sensors is a laser or light stripe sensor, which sweeps a plane of light across the scene or object while observing it from an offset viewpoint, as shown in Figure 12.5b (Rioux and Bird 1993; Curless and Levoy 1995). As the stripe falls across the object, it deforms its shape according to the shape of the surface it is illuminating. It is then a simple matter of using *optical triangulation* to estimate the 3D locations of all the points seen in a particular stripe. In more detail, knowledge of the 3D plane equation of the light stripe allows us to infer the 3D location corresponding to each illuminated pixel, as previously discussed in (2.70–2.71). The accuracy of light striping techniques can be improved by finding the exact temporal peak in illumination for each pixel (Curless and Levoy 1995). The final accuracy of a scanner can be determined using slant edge modulation techniques, i.e., by imaging sharp creases in a calibration object (Goesele, Fuchs, and Seidel 2003).

An interesting variant on light stripe rangefinding is presented by Bouguet and Perona (1999). Instead of projecting a light stripe, they simply wave a stick casting a shadow over a scene or object illuminated by a point light source such as a lamp or the sun (Figure 12.6a). As the shadow falls across two background planes whose orientation relative to the camera is known (or inferred during pre-calibration), the plane equation for each stripe can be inferred from the two projected lines, whose 3D equations are known (Figure 12.6b). The deformation of the shadow as it crosses the object being scanned then reveals its 3D shape, as with regular light stripe rangefinding (Exercise 12.2). This technique can also be used to estimate the 3D geometry of a background scene and how its appearance varies as it moves into shadow, in order to cast new shadows onto the scene (Chuang, Goldman, Curless *et al.* 2003) (Section 10.4.3).

The time it takes to scan an object using a light stripe technique is proportional to the number of depth planes used, which is usually comparable to the number of pixels across an image. A much faster scanner can be constructed by turning different projector pixels on

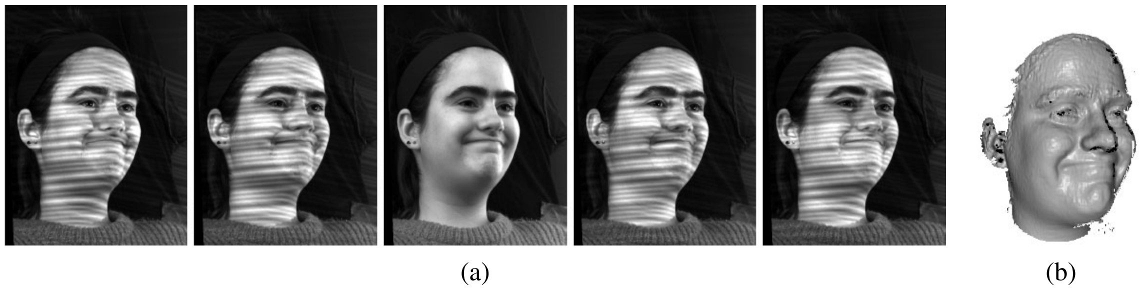


Figure 12.7 Real-time dense 3D face capture using spacetime stereo (Zhang, Snavely, Curless *et al.* 2004) © 2004 ACM: (a) set of five consecutive video frames from one of two stereo cameras (every fifth frame is free of stripe patterns, in order to extract texture); (b) resulting high-quality 3D surface model (depth map visualized as a shaded rendering).

and off in a structured manner, e.g., using a binary or Gray code (Besl 1989). For example, let us assume that the LCD projector we are using has 1024 columns of pixels. Taking the 10-bit binary code corresponding to each column's address $(0 \ldots 1023)$, we project the first bit, then the second, etc. After 10 projections (e.g., a third of a second for a synchronized 30Hz camera-projector system), each pixel in the camera knows which of the 1024 columns of projector light it is seeing. A similar approach can also be used to estimate the refractive properties of an object by placing a monitor behind the object (Zongker, Werner, Curless *et al.* 1999; Chuang, Zongker, Hindorff *et al.* 2000) (Section 13.4). Very fast scanners can also be constructed with a single laser beam, i.e., a real-time *flying spot* optical triangulation scanner (Rioux, Bechthold, Taylor *et al.* 1987).

If even faster, i.e., frame-rate, scanning is required, we can project a single textured pattern into the scene. Proesmans, Van Gool, and Defoort (1998) describe a system where a checkerboard grid is projected onto an object (e.g., a person's face) and the deformation of the grid is used to infer 3D shape. Unfortunately, such a technique only works if the surface is continuous enough to link all of the grid points.

A much better system can be constructed using high-speed custom illumination and sensing hardware. Iddan and Yahav (2001) describe the construction of their 3DV Zcam video-rate depth sensing camera, which projects a pulsed plane of light onto the scene and then integrates the returning light for a short interval, essentially obtaining time-of-flight measurement for the distance to individual pixels in the scene. A good description of earlier time-of-flight systems, including amplitude and frequency modulation schemes for LIDAR, can be found in (Besl 1989).

Instead of using a single camera, it is also possible to construct an active illumination range sensor using stereo imaging setups. The simplest way to do this is to just project random stripe patterns onto the scene to create synthetic texture, which helps match textureless surfaces (Kang, Webb, Zitnick *et al.* 1995). Projecting a known series of stripes, just as in coded pattern single-camera rangefinding, makes the correspondence between pixels unambiguous and allows for the recovery of depth estimates at pixels only seen in a single camera (Scharstein and Szeliski 2003). This technique has been used to produce large numbers of highly accurate registered multi-image stereo pairs and depth maps for the purpose of evaluating stereo correspondence algorithms (Scharstein and Szeliski 2002; Hirschmüller and

Scharstein 2009) and learning depth map priors and parameters (Scharstein and Pal 2007).

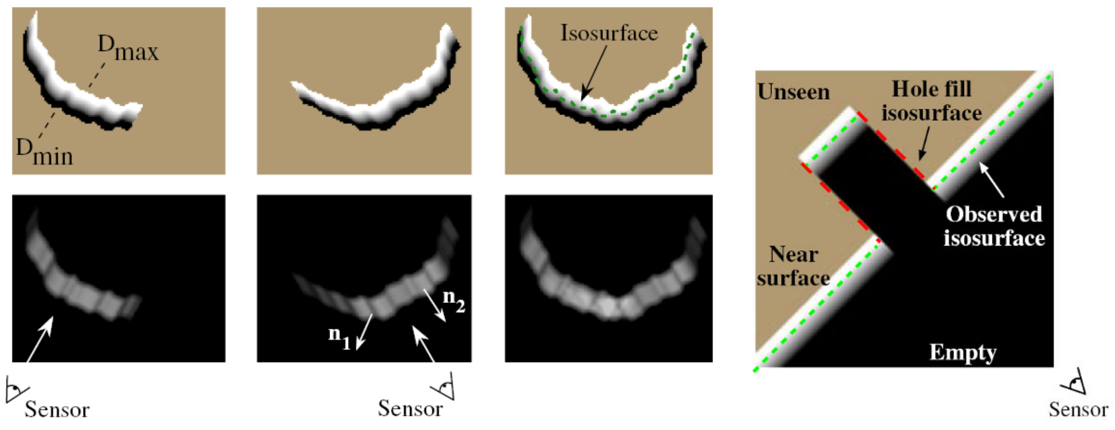
While projecting multiple patterns usually requires the scene or object to remain still, additional processing can enable the production of real-time depth maps for dynamic scenes. The basic idea (Davis, Ramamoorthi, and Rusinkiewicz 2003; Zhang, Curless, and Seitz 2003) is to assume that depth is nearly constant within a 3D space–time window around each pixel and to use the 3D window for matching and reconstruction. Depending on the surface shape and motion, this assumption may be error-prone, as shown in (Davis, Nahab, Ramamoorthi *et al.* 2005). To model shapes more accurately, Zhang, Curless, and Seitz (2003) model the linear disparity variation within the space–time window and show that better results can be obtained by globally optimizing disparity and disparity gradient estimates over video volumes (Zhang, Snavely, Curless *et al.* 2004). Figure 12.7 shows the results of applying this system to a person's face; the frame-rate 3D surface model can then be used for further model-based fitting and computer graphics manipulation (Section 12.6.2).

12.2.1 Range data merging

While individual range images can be useful for applications such as real-time z-keying or facial motion capture, they are often used as building blocks for more complete 3D object modeling. In such applications, the next two steps in processing are the registration (alignment) of partial 3D surface models and their integration into coherent 3D surfaces (Curless 1999). If desired, this can be followed by a model fitting stage using either parametric representations such as generalized cylinders (Agin and Binford 1976; Nevatia and Binford 1977; Marr and Nishihara 1978; Brooks 1981), superquadrics (Pentland 1986; Solina and Bajcsy 1990; Terzopoulos and Metaxas 1991), or non-parametric models such as triangular meshes (Boissonat 1984) or physically-based models (Terzopoulos, Witkin, and Kass 1988; Delingette, Hebert, and Ikeuichi 1992; Terzopoulos and Metaxas 1991; McInerney and Terzopoulos 1993; Terzopoulos 1999). A number of techniques have also been developed for segmenting range images into simpler constituent surfaces (Hoover, Jean-Baptiste, Jiang *et al.* 1996).

The most widely used 3D registration technique is the *iterated closest point* (ICP) algorithm, which alternates between finding the closest point matches between the two surfaces being aligned and then solving a 3D *absolute orientation* problem (Section 6.1.5, (6.31–6.32) (Besl and McKay 1992; Chen and Medioni 1992; Zhang 1994; Szeliski and Lavallée 1996; Gold, Rangarajan, Lu *et al.* 1998; David, DeMenthon, Duraiswami *et al.* 2004; Li and Hartley 2007; Enqvist, Josephson, and Kahl 2009).[3] Since the two surfaces being aligned usually only have partial overlap and may also have outliers, robust matching criteria (Section 6.1.4 and Appendix B.3) are typically used. In order to speed up the determination of the closest point, and also to make the distance-to-surface computation more accurate, one of the two point sets (e.g., the current merged model) can be converted into a *signed distance function*, optionally represented using an *octree spline* for compactness (Lavallée and Szeliski 1995). Variants on the basic ICP algorithm can be used to register 3D point sets under non-rigid deformations, e.g., for medical applications (Feldmar and Ayache 1996; Szeliski and Lavallée 1996). Color values associated with the points or range measurements can also be used as part of the registration process to improve robustness (Johnson and Kang 1997; Pulli 1999).

[3] Some techniques, such as the one developed by Chen and Medioni (1992), use local surface tangent planes to make this computation more accurate and to accelerate convergence.



(a) (b)

Figure 12.8 Range data merging (Curless and Levoy 1996) © 1996 ACM: (a) two signed distance functions (top left) are merged with their (weights) bottom left to produce a combined set of functions (right column) from which an isosurface can be extracted (green dashed line); (b) the signed distance functions are combined with empty and unseen space labels to fill holes in the isosurface.

Unfortunately, the ICP algorithm and its variants can only find a locally optimal alignment between 3D surfaces. If this is not known *a priori*, more global correspondence or search techniques, based on local descriptors invariant to 3D rigid transformations, need to be used. An example of such a descriptor is the *spin image*, which is a local circular projection of a 3D surface patch around the local normal axis (Johnson and Hebert 1999). Another (earlier) example is the *splash* representation introduced by Stein and Medioni (1992).

Once two or more 3D surfaces have been aligned, they can be merged into a single model. One approach is to represent each surface using a triangulated mesh and combine these meshes using a process that is sometimes called *zippering* (Soucy and Laurendeau 1992; Turk and Levoy 1994). Another, now more widely used, approach is to compute a signed distance function that fits all of the 3D data points (Hoppe, DeRose, Duchamp *et al.* 1992; Curless and Levoy 1996; Hilton, Stoddart, Illingworth *et al.* 1996; Wheeler, Sato, and Ikeuchi 1998).

Figure 12.8 shows one such approach, the *volumetric range image processing* (VRIP) technique developed by Curless and Levoy (1996), which first computes a weighted signed distance function from each range image and then merges them using a weighted averaging process. To make the representation more compact, run-length coding is used to encode the empty, seen, and varying (signed distance) voxels, and only the signed distance values near each surface are stored.[4] Once the merged signed distance function has been computed, a zero-crossing surface extraction algorithm, such as *marching cubes* (Lorensen and Cline 1987), can be used to recover a meshed surface model. Figure 12.9 shows an example of the complete range data merging and isosurface extraction pipeline.

Volumetric range data merging techniques based on signed distance or characteristic (inside–outside) functions are also widely used to extract smooth well-behaved surfaces from oriented or unoriented sets of points (Hoppe, DeRose, Duchamp *et al.* 1992; Ohtake, Belyaev,

[4] An alternative, even more compact, representation could be to use octrees (Lavallée and Szeliski 1995).

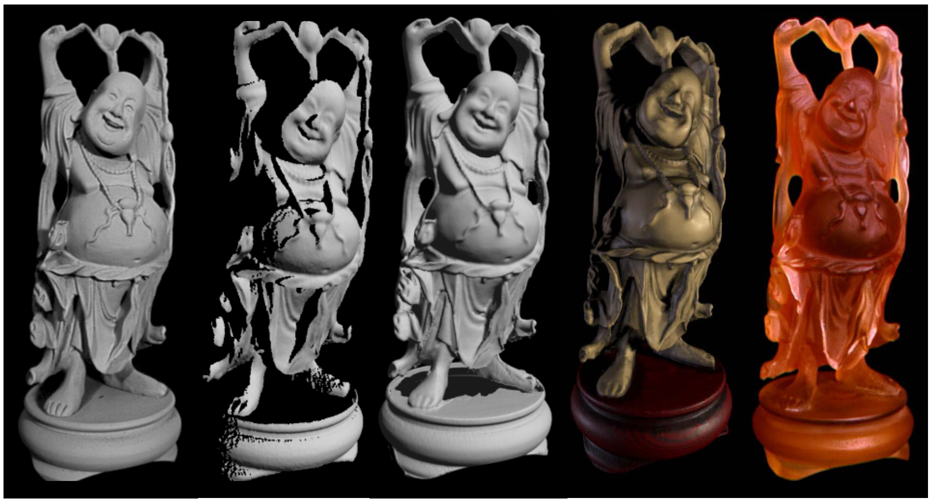


Figure 12.9 Reconstruction and hardcopy of the "Happy Buddha" statuette (Curless and Levoy 1996) © 1996 ACM: (a) photograph of the original statue after spray painting with matte gray; (b) partial range scan; (c) merged range scans; (d) colored rendering of the reconstructed model; (e) hardcopy of the model constructed using stereolithography.

Alexa *et al.* 2003; Kazhdan, Bolitho, and Hoppe 2006; Lempitsky and Boykov 2007; Zach, Pock, and Bischof 2007b; Zach 2008), as discussed in more detail in Section 12.5.1.

12.2.2 *Application*: Digital heritage

Active rangefinding technologies, combined with surface modeling and appearance modeling techniques (Section 12.7), are widely used in the fields of archeological and historical preservation, which often also goes under the name *digital heritage* (MacDonald 2006). In such applications, detailed 3D models of cultural objects are acquired and later used for applications such as analysis, preservation, restoration, and the production of duplicate artwork (Rioux and Bird 1993).

A more recent example of such an endeavor is the Digital Michelangelo project of Levoy, Pulli, Curless *et al.* (2000), which used Cyberware laser stripe scanners and high-quality digital SLR cameras mounted on a large gantry to obtain detailed scans of Michelangelo's David and other sculptures in Florence. The project also took scans of the *Forma Urbis Romae*, an ancient stone map of Rome that had shattered into pieces, for which new matches were obtained using digital techniques. The whole process, from initial planning, to software development, acquisition, and post-processing, took several years (and many volunteers), and produced a wealth of 3D shape and appearance modeling techniques as a result.

Even larger-scale projects are now being attempted, for example, the scanning of complete temple sites such as Angkor-Thom (Ikeuchi and Sato 2001; Ikeuchi and Miyazaki 2007; Banno, Masuda, Oishi *et al.* 2008). Figure 12.10 shows details from this project, including a sample photograph, a detailed 3D (sculptural) head model scanned from ground level, and an aerial overview of the final merged 3D site model, which was acquired using a balloon.

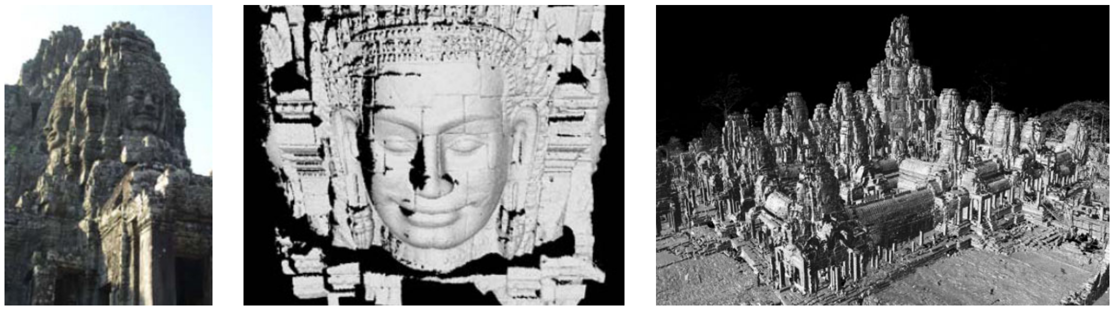


Figure 12.10 Laser range modeling of the Bayon temple at Angkor-Thom (Banno, Masuda, Oishi *et al.* 2008) © 2008 Springer: (a) sample photograph from the site; (b) a detailed head model scanned from the ground; (c) final merged 3D model of the temple scanned using a laser range sensor mounted on a balloon.

12.3 Surface representations

In previous sections, we have seen different representations being used to integrate 3D range scans. We now look at several of these representations in more detail. Explicit surface representations, such as triangle meshes, splines (Farin 1992, 1996), and subdivision surfaces (Stollnitz, DeRose, and Salesin 1996; Zorin, Schröder, and Sweldens 1996; Warren and Weimer 2001; Peters and Reif 2008), enable not only the creation of highly detailed models but also processing operations, such as interpolation (Section 12.3.1), fairing or smoothing, and decimation and simplification (Section 12.3.2). We also examine discrete point-based representations (Section 12.4) and volumetric representations (Section 12.5).

12.3.1 Surface interpolation

One of the most common operations on surfaces is their reconstruction from a set of sparse data constraints, i.e. *scattered data interpolation*. When formulating such problems, surfaces may be parameterized as height fields $f(\boldsymbol{x})$, as 3D parametric surfaces $\boldsymbol{f}(\boldsymbol{x})$, or as non-parametric models such as collections of triangles.

In the section on image processing, we saw how two-dimensional function interpolation and approximation problems $\{d_i\} \rightarrow f(\boldsymbol{x})$ could be cast as energy minimization problems using regularization (Section 3.7.1 (3.94–3.98).[5] Such problems can also specify the locations of discontinuities in the surface as well as local orientation constraints (Terzopoulos 1986b; Zhang, Dugas-Phocion, Samson *et al.* 2002).

One approach to solving such problems is to discretize both the surface and the energy on a discrete grid or mesh using finite element analysis (3.100–3.102) (Terzopoulos 1986b). Such problems can then be solved using sparse system solving techniques, such as multigrid (Briggs, Henson, and McCormick 2000) or hierarchically preconditioned conjugate gradient (Szeliski 2006b). The surface can also be represented using a hierarchical combination of multilevel B-splines (Lee, Wolberg, and Shin 1996).

[5] The difference between interpolation and approximation is that the former requires the surface or function to pass through the data while the latter allows the function to pass near the data, and can therefore be used for surface smoothing as well.

An alternative approach is to use *radial basis* (or *kernel*) functions (Boult and Kender 1986; Nielson 1993). To interpolate a field $\boldsymbol{f}(\boldsymbol{x})$ through (or near) a number of data values $\boldsymbol{d}_i$ located at $\boldsymbol{x}_i$, the *radial basis function* approach uses

$$\boldsymbol{f}(\boldsymbol{x}) = \frac{\sum_i w_i(\boldsymbol{x})\boldsymbol{d}_i}{\sum_i w_i(\boldsymbol{x})}, \tag{12.6}$$

where the weights,

$$w_i(\boldsymbol{x}) = K(\|\boldsymbol{x} - \boldsymbol{x}_i\|), \tag{12.7}$$

are computed using a *radial basis* (spherically symmetrical) function $K(r)$.

If we want the function $\boldsymbol{f}(\boldsymbol{x})$ to exactly interpolate the data points, the kernel functions must either be singular at the origin, $\lim_{r\rightarrow 0} K(r) \rightarrow \infty$ (Nielson 1993), or a dense linear system must be solved to determine the magnitude associated with each basis function (Boult and Kender 1986). It turns out that, for certain regularized problems, e.g., (3.94–3.96), there exist radial basis functions (kernels) that give the same results as a full analytical solution (Boult and Kender 1986). Unfortunately, because the dense system solving is cubic in the number of data points, basis function approaches can only be used for small problems such as feature-based image morphing (Beier and Neely 1992).

When a three-dimensional *parametric surface* is being modeled, the vector-valued function $\boldsymbol{f}$ in (12.6) or (3.94–3.102) encodes 3D coordinates (x, y, z) on the surface and the domain $\boldsymbol{x} = (s, t)$ encodes the surface parameterization. One example of such surfaces are symmetry-seeking parametric models, which are elastically deformable versions of *generalized cylinders*[6] (Terzopoulos, Witkin, and Kass 1987). In these models, s is the parameter *along* the spine of the deformable tube and t is the parameter *around* the tube. A variety of smoothness and radial symmetry forces are used to constrain the model while it is fitted to image-based silhouette curves.

It is also possible to define *non-parametric* surface models such as general triangulated meshes and to equip such meshes (using finite element analysis) with both internal smoothness metrics and external data fitting metrics (Sander and Zucker 1990; Fua and Sander 1992; Delingette, Hebert, and Ikeuichi 1992; McInerney and Terzopoulos 1993). While most of these approaches assume a standard *elastic* deformation model, which uses quadratic internal smoothness terms, it is also possible to use sub-linear energy models in order to better preserve surface creases (Diebel, Thrun, and Brünig 2006). Triangle meshes can also be augmented with either spline elements (Sullivan and Ponce 1998) or subdivision surfaces (Stollnitz, DeRose, and Salesin 1996; Zorin, Schröder, and Sweldens 1996; Warren and Weimer 2001; Peters and Reif 2008) to produce surfaces with better smoothness control.

Both parametric and non-parametric surface models assume that the topology of the surface is known and fixed ahead of time. For more flexible surface modeling, we can either represent the surface as a collection of oriented points (Section 12.4) or use 3D implicit functions (Section 12.5.1), which can also be combined with elastic 3D surface models (McInerney and Terzopoulos 1993).

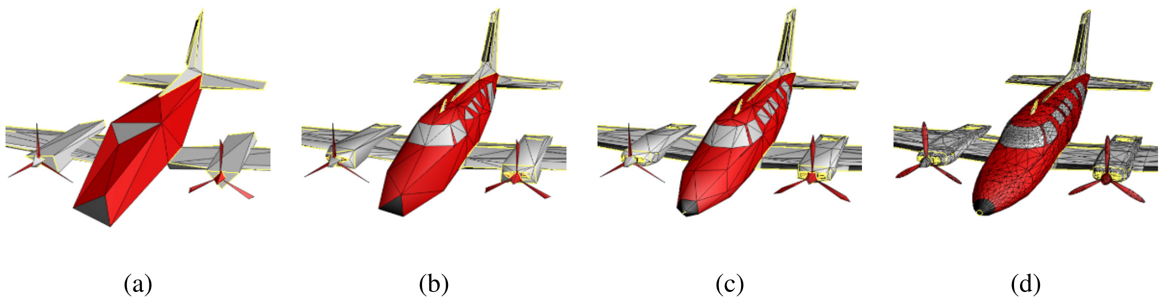


Figure 12.11 Progressive mesh representation of an airplane model (Hoppe 1996) © 1996 ACM: (a) base mesh M^0 (150 faces); (b) mesh M^{175} (500 faces); (c) mesh M^{425} (1000 faces); (d) original mesh $M = M^n$ (13,546 faces).

12.3.2 Surface simplification

Once a triangle mesh has been created from 3D data, it is often desirable to create a hierarchy of mesh models, for example, to control the displayed *level of detail* (LOD) in a computer graphics application. (In essence, this is a 3D analog to image pyramids (Section 3.5).) One approach to doing this is to approximate a given mesh with one that has *subdivision connectivity*, over which a set of triangular wavelet coefficients can then be computed (Eck, DeRose, Duchamp *et al.* 1995). A more continuous approach is to use sequential *edge collapse* operations to go from the original fine-resolution mesh to a coarse base-level mesh (Hoppe 1996). The resulting *progressive mesh* (PM) representation can be used to render the 3D model at arbitrary levels of detail, as shown in Figure 12.11.

12.3.3 Geometry images

While multi-resolution surface representations such as (Eck, DeRose, Duchamp *et al.* 1995; Hoppe 1996) support level of detail operations, they still consist of an irregular collection of triangles, which makes them more difficult to compress and store in a cache-efficient manner.[7]

To make the triangulation completely regular (uniform and gridded), Gu, Gortler, and Hoppe (2002) describe how to create *geometry images* by cutting surface meshes along well-chosen lines and "flattening" the resulting representation into a square. Figure 12.12a shows the resulting (x, y, z) values of the surface mesh mapped over the unit square, while Figure 12.12b shows the associated (n_x, n_y, n_z) *normal map*, i.e., the surface normals associated with each mesh vertex, which can be used to compensate for loss in visual fidelity if the original geometry image is heavily compressed.

[6] A generalized cylinder (Brooks 1981) is a *solid of revolution*, i.e., the result of rotating a (usually smooth) curve around an axis. It can also be generated by sweeping a slowly varying circular cross-section along the axis. (These two interpretations are equivalent.)

[7] Subdivision triangulations, such as those in (Eck, DeRose, Duchamp *et al.* 1995), are *semi-regular*, i.e., regular (ordered and nested) within each subdivided base triangle.

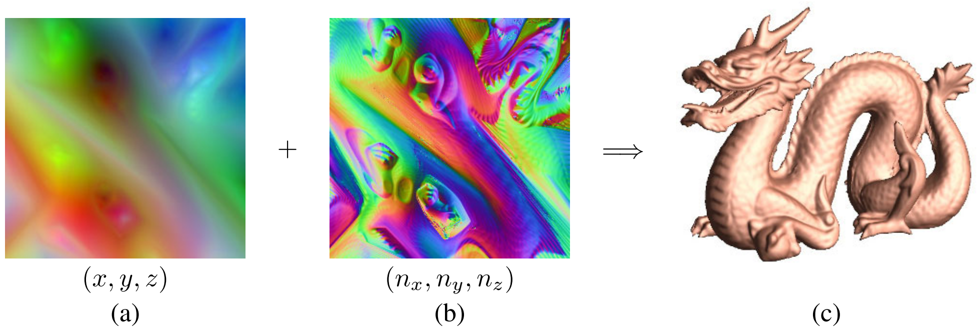


Figure 12.12 Geometry images (Gu, Gortler, and Hoppe 2002) © 2002 ACM: (a) the 257×257 geometry image defines a mesh over the surface; (b) the 512×512 normal map defines vertex normals; (c) final lit 3D model.

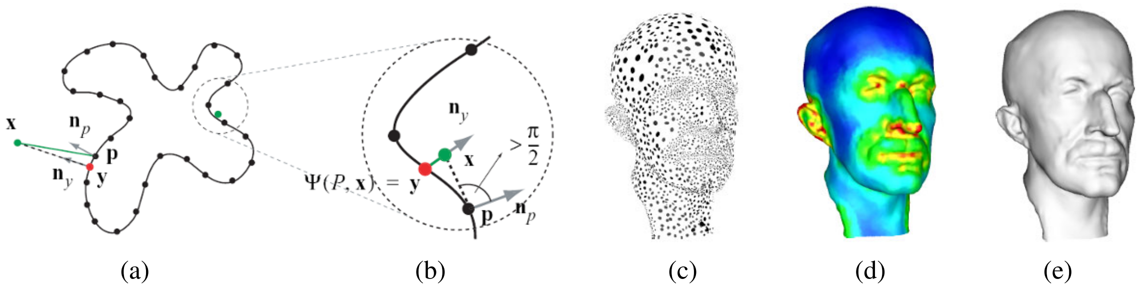


Figure 12.13 Point-based surface modeling with moving least squares (MLS) (Pauly, Keiser, Kobbelt *et al.* 2003) © 2003 ACM: (a) a set of points (black dots) is turned into an implicit inside–outside function (black curve); (b) the signed distance to the nearest oriented point can serve as an approximation to the inside–outside distance; (c) a set of oriented points with variable sampling density representing a 3D surface (head model); (d) local estimate of sampling density, which is used in the moving least squares; (e) reconstructed continuous 3D surface.

12.4 Point-based representations

As we mentioned previously, triangle-based surface models assume that the topology (and often the rough shape) of the 3D model is known ahead of time. While it is possible to re-mesh a model as it is being deformed or fitted, a simpler solution is to dispense with an explicit triangle mesh altogether and to have triangle vertices behave as *oriented points*, or particles, or *surface elements* (surfels) (Szeliski and Tonnesen 1992).

In order to endow the resulting particle system with internal smoothness constraints, pairwise interaction potentials can be defined that approximate the equivalent elastic bending energies that would be obtained using local finite-element analysis.[8] Instead of defining the finite element neighborhood for each particle (vertex) ahead of time, a soft influence function is used to couple nearby particles. The resulting 3D model can change both topology and particle density as it evolves and can therefore be used to interpolate partial 3D data with holes (Szeliski, Tonnesen, and Terzopoulos 1993b). Discontinuities in both the surface orientation and crease curves can also be modeled (Szeliski, Tonnesen, and Terzopoulos 1993a).

[8] As mentioned before, an alternative is to use *sub-linear* interaction potentials, which encourage the preservation of surface creases (Diebel, Thrun, and Brünig 2006).

To render the particle system as a continuous surface, local dynamic triangulation heuristics (Szeliski and Tonnesen 1992) or direct surface element *splatting* (Pfister, Zwicker, van Baar *et al.* 2000) can be used. Another alternative is to first convert the point cloud into an implicit signed distance or inside–outside function, using either minimum signed distances to the oriented points (Hoppe, DeRose, Duchamp *et al.* 1992) or by interpolating a characteristic (inside–outside) function using radial basis functions (Turk and O'Brien 2002; Dinh, Turk, and Slabaugh 2002). Even greater precision over the implicit function fitting, including the ability to handle irregular point densities, can be obtained by computing a *moving least squares* (MLS) estimate of the signed distance function (Alexa, Behr, Cohen-Or *et al.* 2003; Pauly, Keiser, Kobbelt *et al.* 2003), as shown in Figure 12.13. Further improvements can be obtained using local sphere fitting (Guennebaud and Gross 2007), faster and more accurate re-sampling (Guennebaud, Germann, and Gross 2008), and kernel regression to better tolerate outliers (Oztireli, Guennebaud, and Gross 2008).

12.5 Volumetric representations

A third alternative for modeling 3D surfaces is to construct 3D volumetric inside–outside functions. We already saw examples of this in Section 11.6.1, where we looked at voxel coloring (Seitz and Dyer 1999), space carving (Kutulakos and Seitz 2000), and level set (Faugeras and Keriven 1998; Pons, Keriven, and Faugeras 2007) techniques for stereo matching, and Section 11.6.2, where we discussed using binary silhouette images to reconstruct volumes.

In this section, we look at continuous *implicit* (inside–outside) functions to represent 3D shape.

12.5.1 Implicit surfaces and level sets

While polyhedral and voxel-based representations can represent three-dimensional shapes to an arbitrary precision, they lack some of the intrinsic smoothness properties available with continuous implicit surfaces, which use an *indicator function* (*characteristic function*) $F(x, y, z)$ to indicate which 3D points are inside $F(x, y, z) < 0$ or outside $F(x, y, z) > 0$ the object.

An early example of using implicit functions to model 3D objects in computer vision are superquadrics, which are a generalization of quadric (e.g., ellipsoidal) parametric volumetric models,

$$F(x, y, z) = \left(\left(\frac{x}{a_1} \right)^{2/\epsilon_2} + \left(\frac{y}{a_2} \right)^{2/\epsilon_2} \right)^{\epsilon_2/\epsilon_1} + \left(\frac{x}{a_1} \right)^{2/\epsilon_1} - 1 = 0 \tag{12.8}$$

(Pentland 1986; Solina and Bajcsy 1990; Waithe and Ferrie 1991; Leonardis, Jaklič, and Solina 1997). The values of (a_1, a_2, a_3) control the extent of model along each (x, y, z) axis, while the values of (ϵ_1, ϵ_2) control how "square" it is. To model a wider variety of shapes, superquadrics are usually combined with either rigid or non-rigid deformations (Terzopoulos and Metaxas 1991; Metaxas and Terzopoulos 2002). Superquadric models can either be fit to range data or used directly for stereo matching.

A different kind of implicit shape model can be constructed by defining a *signed distance function* over a regular three-dimensional grid, optionally using an octree spline to represent this function more coarsely away from its surface (zero-set) (Lavallée and Szeliski 1995; Szeliski and Lavallée 1996; Frisken, Perry, Rockwood *et al.* 2000; Ohtake, Belyaev, Alexa *et al.* 2003). We have already seen examples of signed distance functions being used to represent distance transforms (Section 3.3.3), level sets for 2D contour fitting and tracking (Section 5.1.4), volumetric stereo (Section 11.6.1), range data merging (Section 12.2.1), and point-based modeling (Section 12.4). The advantage of representing such functions directly on a grid is that it is quick and easy to look up distance function values for any (x, y, z) location and also easy to extract the isosurface using the marching cubes algorithm (Lorensen and Cline 1987). The work of Ohtake, Belyaev, Alexa *et al.* (2003) is particularly notable since it allows for several distance functions to be used simultaneously and then combined locally to produce sharp features such as creases.

Poisson surface reconstruction (Kazhdan, Bolitho, and Hoppe 2006) uses a closely related volumetric function, namely a smoothed 0/1 inside–outside (characteristic) function, which can be thought of as a clipped signed distance function. The gradients for this function are set to lie along oriented surface normals near known surface points and 0 elsewhere. The function itself is represented using a quadratic tensor-product B-spline over an octree, which provides a compact representation with larger cells away from the surface or in regions of lower point density, and also admits the efficient solution of the related Poisson equations (3.100–3.102), see Section 9.3.4 (Pérez, Gangnet, and Blake 2003).

It is also possible to replace the quadratic penalties used in the Poisson equations with L_1 (total variation) constraints and still obtain a convex optimization problem, which can be solved using either continuous (Zach, Pock, and Bischof 2007b; Zach 2008) or discrete graph cut (Lempitsky and Boykov 2007) techniques.

Signed distance functions also play an integral role in level-set evolution equations ((Sections 5.1.4 and 11.6.1), where the values of distance transforms on the mesh are updated as the surface evolves to fit multi-view stereo photoconsistency measures (Faugeras and Keriven 1998).

12.6 Model-based reconstruction

When we know something ahead of time about the objects we are trying to model, we can construct more detailed and reliable 3D models using specialized techniques and representations. For example, architecture is usually made up of large planar regions and other parametric forms (such as surfaces of revolution), usually oriented perpendicular to gravity and to each other (Section 12.6.1). Heads and faces can be represented using low-dimensional, non-rigid shape models, since the variability in shape and appearance of human faces, while extremely large, is still bounded (Section 12.6.2). Human bodies or parts, such as hands, form highly articulated structures, which can be represented using kinematic chains of piecewise rigid skeletal elements linked by joints (Section 12.6.4).

In this section, we highlight some of the main ideas, representations, and modeling algorithms used for these three cases. Additional details and references can be found in specialized conferences and workshops devoted to these topics, e.g., the International Symposium on 3D Data Processing, Visualization, and Transmission (3DPVT), the International Conference on 3D Digital Imaging and Modeling (3DIM), the International Conference on Automatic Face and Gesture Recognition (FG), the IEEE Workshop on Analysis and Modeling of Faces

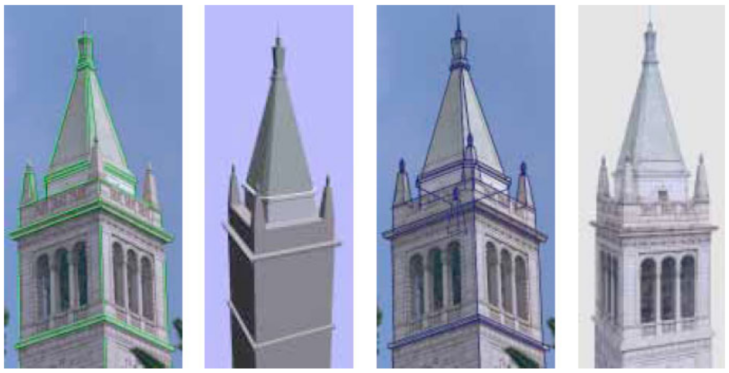


Figure 12.14 Interactive architectural modeling using the Façade system (Debevec, Taylor, and Malik 1996) © 1996 ACM: (a) input image with user-drawn edges shown in green; (b) shaded 3D solid model; (c) geometric primitives overlaid onto the input image; (d) final view-dependent, texture-mapped 3D model.

and Gestures, and the International Workshop on Tracking Humans for the Evaluation of their Motion in Image Sequences (THEMIS).

12.6.1 Architecture

Architectural modeling, especially from aerial photography, has been one of the longest studied problems in both photogrammetry and computer vision (Walker and Herman 1988). Recently, the development of reliable image-based modeling techniques, as well as the prevalence of digital cameras and 3D computer games, has spurred renewed interest in this area.

The work by Debevec, Taylor, and Malik (1996) was one of the earliest hybrid geometry- and image-based modeling and rendering systems. Their Façade system combines an interactive image-guided geometric modeling tool with model-based (local plane plus parallax) stereo matching and view-dependent texture mapping. During the interactive photogrammetric modeling phase, the user selects block elements and aligns their edges with visible edges in the input images (Figure 12.14a). The system then automatically computes the dimensions and locations of the blocks along with the camera positions using constrained optimization (Figure 12.14b–c). This approach is intrinsically more reliable than general feature-based structure from motion, because it exploits the strong geometry available in the block primitives. Related work by Becker and Bove (1995), Horry, Anjyo, and Arai (1997), and Criminisi, Reid, and Zisserman (2000) exploits similar information available from vanishing points. In the interactive, image-based modeling system of Sinha, Steedly, Szeliski *et al.* (2008), vanishing point directions are used to guide the user drawing of polygons, which are then automatically fitted to sparse 3D points recovered using structure from motion.

Once the rough geometry has been estimated, more detailed offset maps can be computed for each planar face using a local plane sweep, which Debevec, Taylor, and Malik (1996) call *model-based stereo*. Finally, during rendering, images from different viewpoints are warped and blended together as the camera moves around the scene, using a process (related to light field and Lumigraph rendering, see Section 13.3) called *view-dependent texture mapping* (Figure 12.14d).

For interior modeling, instead of working with single pictures, it is more useful to work

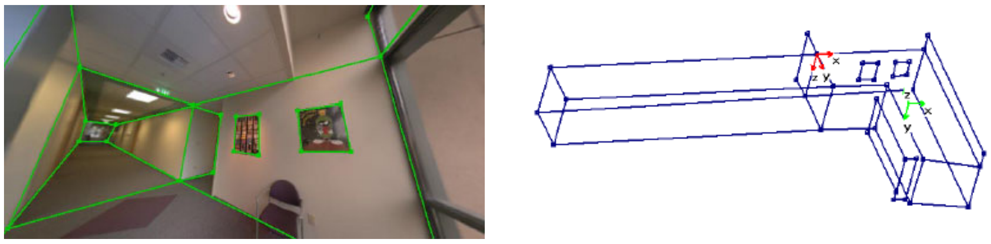


Figure 12.15 Interactive 3D modeling from panoramas (Shum, Han, and Szeliski 1998) © 1998 IEEE: (a) wide-angle view of a panorama with user-drawn vertical and horizontal (axis-aligned) lines; (b) single-view reconstruction of the corridors.

with panoramas, since you can see larger extents of walls and other structures. The 3D modeling system developed by Shum, Han, and Szeliski (1998) first constructs calibrated panoramas from multiple images (Section 7.4) and then has the user draw vertical and horizontal lines in the image to demarcate the boundaries of planar regions. The lines are initially used to establish an absolute rotation for each panorama and are later used (along with the inferred vertices and planes) to optimize the 3D structure, which can be recovered up to scale from one or more images (Figure 12.15). 360° high dynamic range panoramas can also be used for outdoor modeling, since they provide highly reliable estimates of relative camera orientations as well as vanishing point directions (Antone and Teller 2002; Teller, Antone, Bodnar *et al.* 2003).

While earlier image-based modeling systems required some user authoring, Werner and Zisserman (2002) present a fully automated line-based reconstruction system. As described in Section 7.5.1, they first detect lines and vanishing points and use them to calibrate the camera; then they establish line correspondences using both appearance matching and trifocal tensors, which enables them to reconstruct families of 3D line segments, as shown in Figure 12.16a. They then generate plane hypotheses, using both co-planar 3D lines and a plane sweep (Section 11.1.2) based on cross-correlation scores evaluated at interest points. Intersections of planes are used to determine the extent of each plane, i.e., an initial coarse geometry, which is then refined with the addition of rectangular or wedge-shaped indentations and extrusions (Figure 12.16c). Note that when top-down maps of the buildings being modeled are available, these can be used to further constrain the 3D modeling process (Robertson and Cipolla 2002, 2009). The idea of using matched 3D lines for estimating vanishing point directions and dominant planes continues to be used in a number of recent fully automated image-based architectural modeling systems (Zebedin, Bauer, Karner *et al.* 2008; Mičušík and Košecká 2009; Furukawa, Curless, Seitz *et al.* 2009b; Sinha, Steedly, and Szeliski 2009).

Another common characteristic of architecture is the repeated use of primitives such as windows, doors, and colonnades. Architectural modeling systems can be designed to search for such repeated elements and to use them as part of the structure inference process (Dick, Torr, and Cipolla 2004; Mueller, Zeng, Wonka *et al.* 2007; Schindler, Krishnamurthy, Lublinerman *et al.* 2008; Sinha, Steedly, Szeliski *et al.* 2008).

The combination of all these techniques now makes it possible to reconstruct the structure of large 3D scenes (Zhu and Kanade 2008). For example, the *Urbanscan* system of Polle-

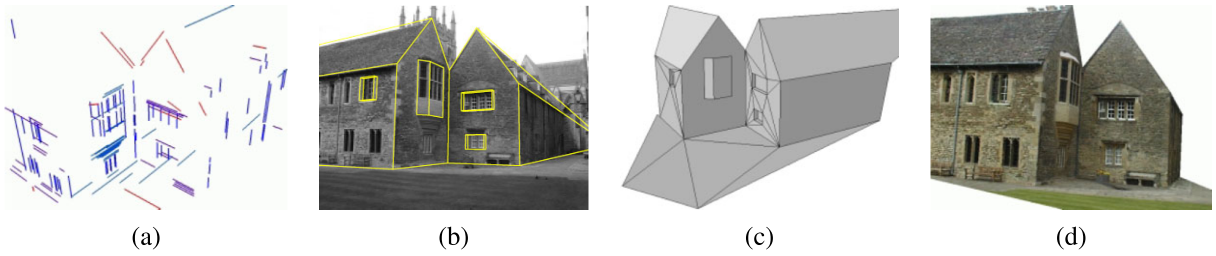


Figure 12.16 Automated architectural reconstruction using 3D lines and planes (Werner and Zisserman 2002) © 2002 Springer: (a) reconstructed 3D lines, color coded by their vanishing directions; (b) wire-frame model superimposed onto an input image; (c) triangulated piecewise-planar model with windows; (d) final texture-mapped model.

feys, Nistér, Frahm *et al.* (2008) reconstructs texture-mapped 3D models of city streets from videos acquired with a GPS-equipped vehicle. To obtain real-time performance, they use both optimized on-line structure-from-motion algorithms, as well as GPU implementations of plane-sweep stereo aligned to dominant planes and depth map fusion. Cornelis, Leibe, Cornelis *et al.* (2008) present a related system that also uses plane-sweep stereo (aligned to vertical building façades) combined with object recognition and segmentation for vehicles. Mičušík and Košecká (2009) build on these results using omni-directional images and super-pixel-based stereo matching along dominant plane orientations. Reconstruction directly from active range scanning data combined with color imagery that has been compensated for exposure and lighting variations is also possible (Chen and Chen 2008; Stamos, Liu, Chen *et al.* 2008; Troccoli and Allen 2008).

12.6.2 Heads and faces

Another area in which specialized shape and appearance models are extremely helpful is in the modeling of heads and faces. Even though the appearance of people seems at first glance to be infinitely variable, the actual shape of a person's head and face can be described reasonably well using a few dozen parameters (Pighin, Hecker, Lischinski *et al.* 1998; Guenter, Grimm, Wood *et al.* 1998; DeCarlo, Metaxas, and Stone 1998; Blanz and Vetter 1999; Shan, Liu, and Zhang 2001).

Figure 12.17 shows an example of an image-based modeling system, where user-specified keypoints in several images are used to fit a generic head model to a person's face. As you can see in Figure 12.17c, after specifying just over 100 keypoints, the shape of the face has become quite adapted and recognizable. Extracting a texture map from the original images and then applying it to the head model results in an animatable model with striking visual fidelity (Figure 12.18a).

A more powerful system can be built by applying *principal component analysis* (PCA) to a collection of 3D scanned faces, which is a topic we discuss in Section 12.6.3. As you can see in Figure 12.19, it is then possible to fit morphable 3D models to single images and to use such models for a variety of animation and visual effects (Blanz and Vetter 1999). It is also possible to design stereo matching algorithms that optimize directly for the head model parameters (Shan, Liu, and Zhang 2001; Kang and Jones 2002) or to use the output of real-

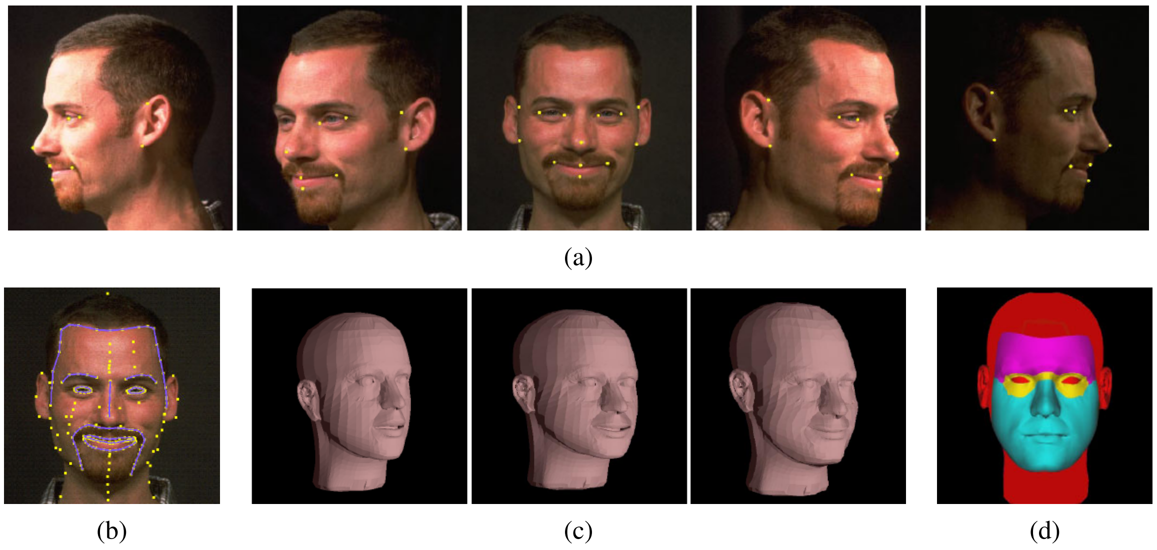


Figure 12.17 3D model fitting to a collection of images: (Pighin, Hecker, Lischinski *et al.* 1998) © 1998 ACM: (a) set of five input images along with user-selected keypoints; (b) the complete set of keypoints and curves; (c) three meshes—the original, adapted after 13 keypoints, and after an additional 99 keypoints; (d) the partition of the image into separately animatable regions.

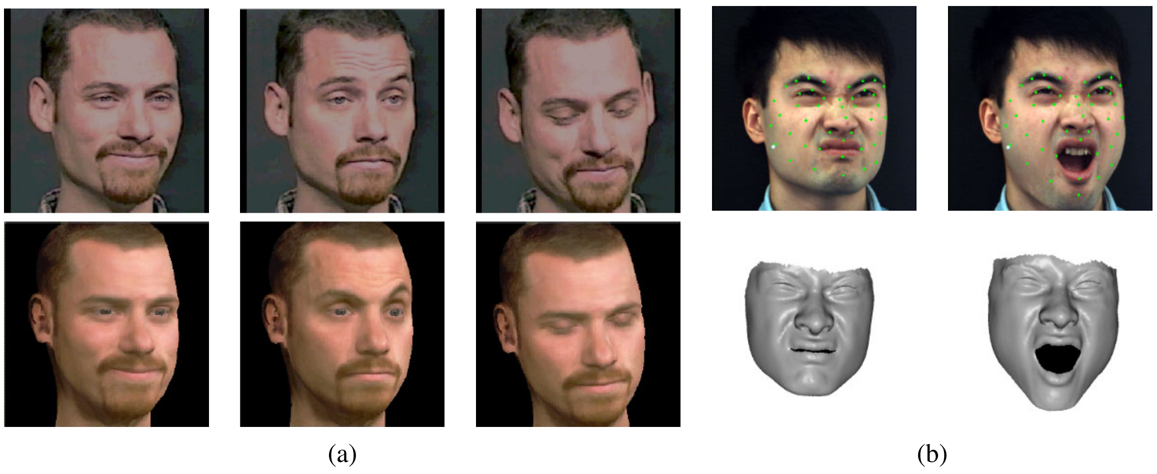


Figure 12.18 Head and expression tracking and re-animation using deformable 3D models. (a) Models fit directly to five input video streams (Pighin, Szeliski, and Salesin 2002) © 2002 Springer: The bottom row shows the results of re-animating a synthetic texture-mapped 3D model with pose and expression parameters fitted to the input images in the top row. (b) Models fit to frame-rate spacetime stereo surface models (Zhang, Snavely, Curless *et al.* 2004) © 2004 ACM: The top row shows the input images with synthetic green markers overlaid, while the bottom row shows the fitted 3D surface model.

time stereo with active illumination (Zhang, Snavely, Curless *et al.* 2004) (Figures 12.7 and 12.18b).

As the sophistication of 3D facial capture systems evolves, so does the detail and realism in the reconstructed models. Newer systems can capture (in real-time) not only surface details such as wrinkles and creases, but also accurate models of skin reflection, translucency, and sub-surface scattering (Weyrich, Matusik, Pfister *et al.* 2006; Golovinskiy, Matusik, ster *et al.* 2006; Bickel, Botsch, Angst *et al.* 2007; Igarashi, Nishino, and Nayar 2007).

Once a 3D head model has been constructed, it can be used in a variety of applications, such as head tracking (Toyama 1998; Lepetit, Pilet, and Fua 2004; Matthews, Xiao, and Baker 2007), as shown in Figures 4.29 and 14.24, and face transfer, i.e., replacing one person's face with another in a video (Bregler, Covell, and Slaney 1997; Vlasic, Brand, Pfister *et al.* 2005). Additional applications include face beautification by warping face images toward a more attractive "standard" (Leyvand, Cohen-Or, Dror *et al.* 2008), face de-identification for privacy protection (Gross, Sweeney, De la Torre *et al.* 2008), and face swapping (Bitouk, Kumar, Dhillon *et al.* 2008).

12.6.3 *Application*: Facial animation

Perhaps the most widely used application of 3D head modeling is facial animation. Once a parameterized 3D model of shape and appearance (surface texture) has been constructed, it can be used directly to track a person's facial motions (Figure 12.18a) and to animate a different character with these same motions and expressions (Pighin, Szeliski, and Salesin 2002).

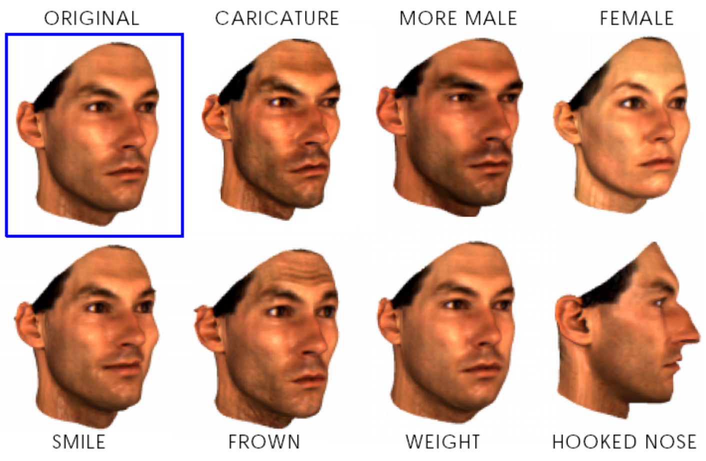
An improved version of such a system can be constructed by first applying principal component analysis (PCA) to the space of possible head shapes and facial appearances. Blanz and Vetter (1999) describe a system where they first capture a set of 200 colored range scans of faces (Figure 12.19a), which can be represented as a large collection of (X, Y, Z, R, G, B) samples (vertices).[9] In order for 3D morphing to be meaningful, corresponding vertices in different people's scans must first be put into correspondence (Pighin, Hecker, Lischinski *et al.* 1998). Once this is done, PCA can be applied to more naturally parameterize the 3D morphable model. The flexibility of this model can be increased by performing separate analyses in different subregions, such as the eyes, nose, and mouth, just as in modular eigenspaces (Moghaddam and Pentland 1997).

After computing a subspace representation, different directions in this space can be associated with different characteristics such as gender, facial expressions, or facial features (Figure 12.19a). As in the work of Rowland and Perrett (1995), faces can be turned into caricatures by exaggerating their displacement from the mean image.

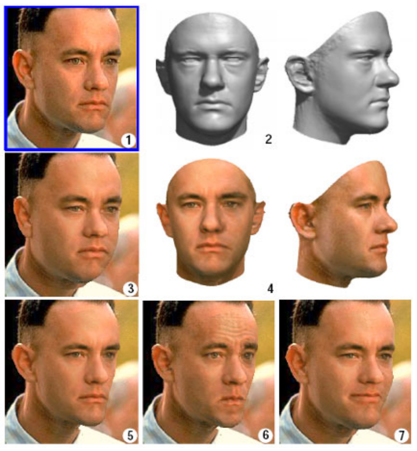
3D morphable models can be fitted to a single image using gradient descent on the error between the input image and the re-synthesized model image, after an initial manual placement of the model in an approximately correct pose, scale, and location (Figures 12.19b–c). The efficiency of this fitting process can be increased using inverse compositional image alignment (8.64–8.65), as described by Romdhani and Vetter (2003).

The resulting texture-mapped 3D model can then be modified to produce a variety of vi-

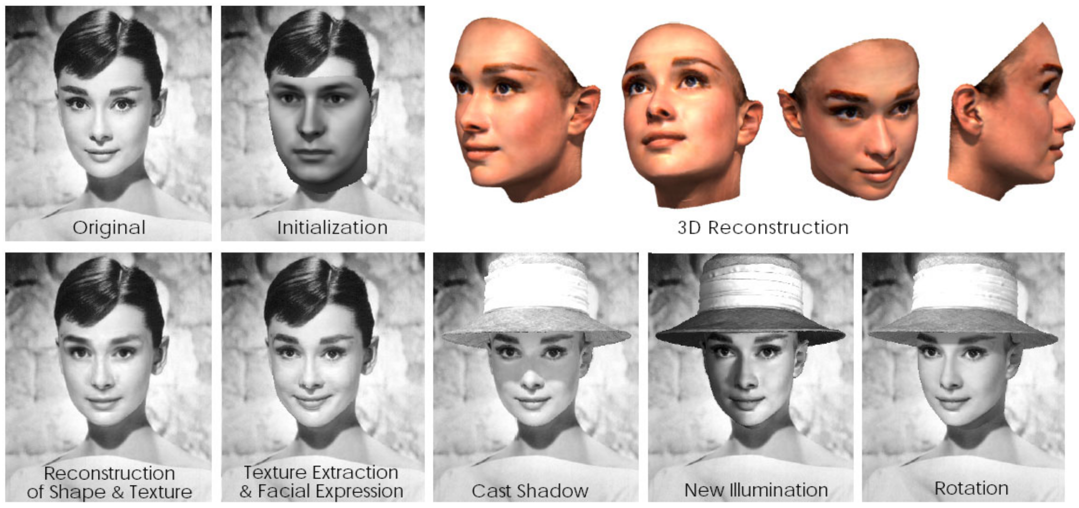
[9] A cylindrical coordinate system provides a natural two-dimensional embedding for this collection, but such an embedding is not necessary to perform PCA.



(a)



(b)



(c)

Figure 12.19 3D morphable face model (Blanz and Vetter 1999) © 1999 ACM: (a) original 3D face model with the addition of shape and texture variations in specific directions: deviation from the mean (caricature), gender, expression, weight, and nose shape; (b) a 3D morphable model is fit to a single image, after which its weight or expression can be manipulated; (c) another example of a 3D reconstruction along with a different set of 3D manipulations such as lighting and pose change.

sual effects, including changing a person's weight or expression, or three-dimensional effects such as re-lighting or 3D video-based animation (Section 13.5.1). Such models can also be used for video compression, e.g., by only transmitting a small number of facial expression and pose parameters to drive a synthetic avatar (Eisert, Wiegand, and Girod 2000; Gao, Chen, Wang *et al.* 2003).

3D facial animation is often matched to the performance of an actor, in what is known as *performance-driven animation* (Section 4.1.5) (Williams 1990). Traditional performance-driven animation systems use marker-based motion capture (Ma, Jones, Chiang *et al.* 2008), while some newer systems use video footage to control the animation (Buck, Finkelstein, Jacobs *et al.* 2000; Pighin, Szeliski, and Salesin 2002; Zhang, Snavely, Curless *et al.* 2004; Vlasic, Brand, Pfister *et al.* 2005).

An example of the latter approach is the system developed for the film *Benjamin Button*, in which Digital Domain used the CONTOUR system from Mova[10] to capture actor Brad Pitt's facial motions and expressions (Roble and Zafar 2009). CONTOUR uses a combination of phosphorescent paint and multiple high-resolution video cameras to capture real-time 3D range scans of the actor. These 3D models were then translated into Facial Action Coding System (FACS) shape and expression parameters (Ekman and Friesen 1978) to drive a different (older) synthetically animated computer-generated imagery (CGI) character.

12.6.4 Whole body modeling and tracking

The topics of tracking humans, modeling their shape and appearance, and recognizing their activities, are some of the most actively studied areas of computer vision. Annual conferences[11] and special journal issues (Hilton, Fua, and Ronfard 2006) are devoted to this subject, and two recent surveys (Forsyth, Arikan, Ikemoto *et al.* 2006; Moeslund, Hilton, and Krüger 2006) each list over 400 papers devoted to these topics.[12] The HumanEva database of articulated human motions[13] contains multi-view video sequences of human actions along with corresponding motion capture data, evaluation code, and a reference 3D tracker based on particle filtering. The companion paper by Sigal, Balan, and Black (2010) not only describes the database and evaluation but also has a nice survey of important work in this field.

Given the breadth of this area, it is difficult to categorize all of this research, especially since different techniques usually build on each other. Moeslund, Hilton, and Krüger (2006) divide their survey into initialization, tracking (which includes background modeling and segmentation), pose estimation, and action (activity) recognition. Forsyth, Arikan, Ikemoto *et al.* (2006) divide their survey into sections on tracking (background subtraction, deformable templates, flow, and probabilistic models), recovering 3D pose from 2D observations, and data association and body parts. They also include a section on motion synthesis, which is more widely studied in computer graphics (Arikan and Forsyth 2002; Kovar, Gleicher, and Pighin 2002; Lee, Chai, Reitsma *et al.* 2002; Li, Wang, and Shum 2002; Pullen and Bregler

[10] http://www.mova.com.

[11] International Conference on Automatic Face and Gesture Recognition (FG), IEEE Workshop on Analysis and Modeling of Faces and Gestures, and International Workshop on Tracking Humans for the Evaluation of their Motion in Image Sequences (THEMIS).

[12] Older surveys include those by Gavrila (1999) and Moeslund and Granum (2001). Some surveys on gesture recognition, which we do not cover in this book, include those by Pavlović, Sharma, and Huang (1997) and Yang, Ahuja, and Tabb (2002).

[13] http://vision.cs.brown.edu/humaneva/.

2002), see Section 13.5.2. Another potential taxonomy for work in this field would be along the lines of whether 2D or 3D (or multi-view) images are used as input and whether 2D or 3D kinematic models are used.

In this section, we briefly review some of the more seminal and widely cited papers in the areas of background subtraction, initialization and detection, tracking with flow, 3D kinematic models, probabilistic models, adaptive shape modeling, and activity recognition. We refer the reader to the previously mentioned surveys for other topics and more details.

Background subtraction. One of the first steps in many (but certainly not all) human tracking systems is to model the background in order to extract the moving foreground objects (silhouettes) corresponding to people. Toyama, Krumm, Brumitt *et al.* (1999) review several *difference matting* and *background maintenance* (modeling) techniques and provide a good introduction to this topic. Stauffer and Grimson (1999) describe some techniques based on mixture models, while Sidenbladh and Black (2003) develop a more comprehensive treatment, which models not only the background image statistics but also the appearance of the foreground objects, e.g., their edge and motion (frame difference) statistics.

Once silhouettes have been extracted from one or more cameras, they can then be modeled using deformable templates or other contour models (Baumberg and Hogg 1996; Wren, Azarbayejani, Darrell *et al.* 1997). Tracking such silhouettes over time supports the analysis of multiple people moving around a scene, including building shape and appearance models and detecting if they are carrying objects (Haritaoglu, Harwood, and Davis 2000; Mittal and Davis 2003; Dimitrijevic, Lepetit, and Fua 2006).

Initialization and detection. In order to track people in a fully automated manner, it is necessary to first detect (or re-acquire) their presence in individual video frames. This topic is closely related to *pedestrian detection*, which is often considered as a kind of object recognition (Mori, Ren, Efros *et al.* 2004; Felzenszwalb and Huttenlocher 2005; Felzenszwalb, McAllester, and Ramanan 2008), and is therefore treated in more depth in Section 14.1.2. Additional techniques for initializing 3D trackers based on 2D images include those described by Howe, Leventon, and Freeman (2000), Rosales and Sclaroff (2000), Shakhnarovich, Viola, and Darrell (2003), Sminchisescu, Kanaujia, Li *et al.* (2005), Agarwal and Triggs (2006), Lee and Cohen (2006), Sigal and Black (2006), and Stenger, Thayananthan, Torr *et al.* (2006).

Single-frame human detection and pose estimation algorithms can sometimes be used by themselves to perform tracking (Ramanan, Forsyth, and Zisserman 2005; Rogez, Rihan, Ramalingam *et al.* 2008; Bourdev and Malik 2009), as described in Section 4.1.4. More often, however, they are combined with frame-to-frame tracking techniques to provide better reliability (Fossati, Dimitrijevic, Lepetit *et al.* 2007; Andriluka, Roth, and Schiele 2008; Ferrari, Marin-Jimenez, and Zisserman 2008).

Tracking with flow. The tracking of people and their pose from frame to frame can be enhanced by computing optic flow or matching the appearance of their limbs from one frame to another. For example, the *cardboard people* model of Ju, Black, and Yacoob (1996) models the appearance of each leg portion (upper and lower) as a moving rectangle, and uses optic flow to estimate their location in each subsequent frame. Cham and Rehg (1999) and Sidenbladh, Black, and Fleet (2000) track limbs using optical flow and templates, along with

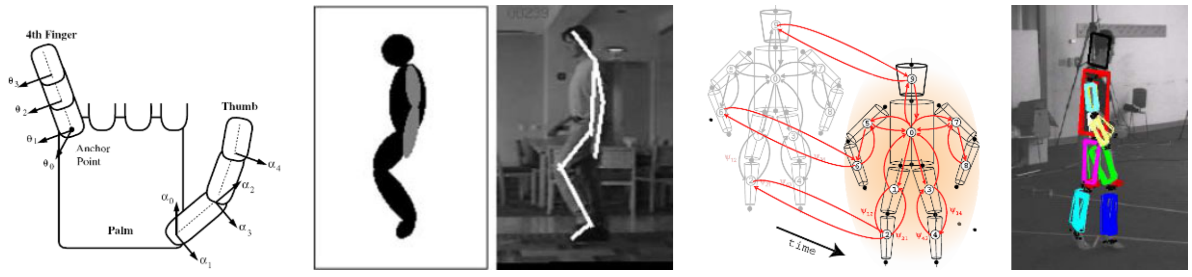


Figure 12.20 Tracking 3D human motion: (a) kinematic chain model for a human hand (Rehg, Morris, and Kanade 2003) © 2003, reprinted by permission of SAGE; (b) tracking a kinematic chain blob model in a video sequence (Bregler, Malik, and Pullen 2004) © 2004 Springer; (c–d) probabilistic loose-limbed collection of body parts (Sigal, Bhatia, Roth *et al.* 2004)

techniques for dealing with multiple hypotheses and uncertainty. Bregler, Malik, and Pullen (2004) use a full 3D model of limb and body motion, as described below. It is also possible to match the estimated motion field itself to some prototypes in order to identify the particular phase of a running motion or to match two low-resolution video portions in order to perform video replacement (Efros, Berg, Mori *et al.* 2003).

3D kinematic models. The effectiveness of human modeling and tracking can be greatly enhanced using a more accurate 3D model of a person's shape and motion. Underlying such representations, which are ubiquitous in 3D computer animation in games and special effects, is a *kinematic model* or *kinematic chain*, which specifies the length of each limb in a skeleton as well as the 2D or 3D rotation angles between the limbs or segments (Figure 12.20a–b). Inferring the values of the joint angles from the locations of the visible surface points is called *inverse kinematics* (IK) and is widely studied in computer graphics.

Figure 12.20a shows the kinematic model for a human hand used by Rehg, Morris, and Kanade (2003) to track hand motion in a video. As you can see, the attachment points between the fingers and the thumb have two degrees of freedom, while the finger joints themselves have only one. Using this kind of model can greatly enhance the ability of an edge-based tracker to cope with rapid motion, ambiguities in 3D pose, and partial occlusions.

Kinematic chain models are even more widely used for whole body modeling and tracking (O'Rourke and Badler 1980; Hogg 1983; Rohr 1994). One popular approach is to associate an ellipsoid or superquadric with each rigid limb in the kinematic model, as shown in Figure 12.20b. This model can then be fitted to each frame in one or more video streams either by matching silhouettes extracted from known backgrounds or by matching and tracking the locations of occluding edges (Gavrila and Davis 1996; Kakadiaris and Metaxas 2000; Bregler, Malik, and Pullen 2004; Kehl and Van Gool 2006). Note that some techniques use 2D models coupled to 2D measurements, some use 3D measurements (range data or multi-view video) with 3D models, and some use monocular video to infer and track 3D models directly.

It is also possible to use temporal models to improve the tracking of periodic motions, such as walking, by analyzing the joint angles as functions of time (Polana and Nelson 1997; Seitz and Dyer 1997; Cutler and Davis 2000). The generality and applicability of such tech-

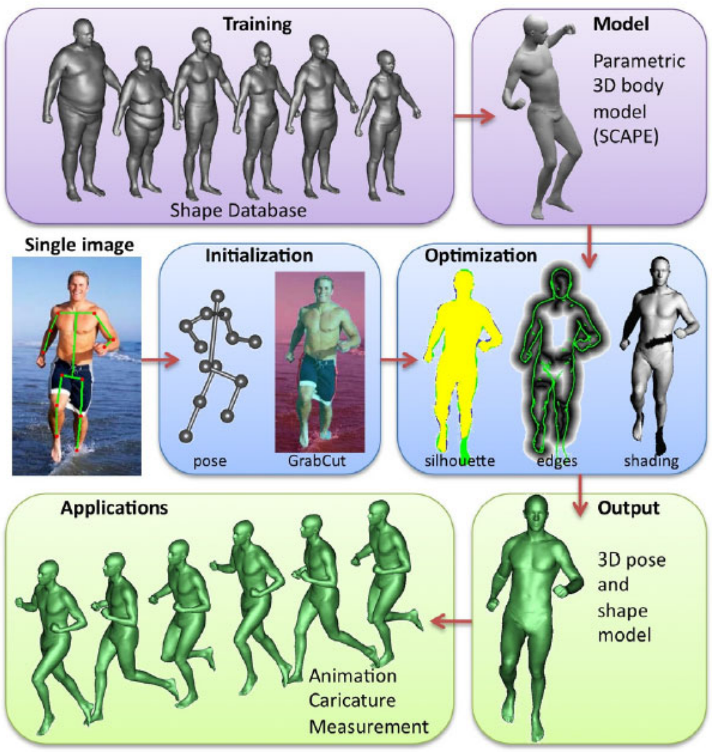


Figure 12.21 Estimating human shape and pose from a single image using a parametric 3D model (Guan, Weiss, Bălan *et al.* 2009) © 2009 IEEE.

niques can be improved by learning typical motion patterns using principal component analysis (Sidenbladh, Black, and Fleet 2000; Urtasun, Fleet, and Fua 2006).

Probabilistic models. Because tracking can be such a difficult task, sophisticated probabilistic inference techniques are often used to estimate the likely states of the person being tracked. One popular approach, called *particle filtering* (Isard and Blake 1998), was originally developed for tracking the outlines of people and hands, as described in Section 5.1.2 (Figures 5.6–5.8). It was subsequently applied to whole-body tracking (Deutscher, Blake, and Reid 2000; Sidenbladh, Black, and Fleet 2000; Deutscher and Reid 2005) and continues to be used in modern trackers (Ong, Micilotta, Bowden *et al.* 2006). Alternative approaches to handling the uncertainty inherent in tracking include multiple hypothesis tracking (Cham and Rehg 1999) and inflated covariances (Sminchisescu and Triggs 2001).

Figure 12.20c–d shows an example of a sophisticated spatio-temporal probabilistic graphical model called *loose-limbed people*, which models not only the geometric relationship between various limbs, but also their likely temporal dynamics (Sigal, Bhatia, Roth *et al.* 2004). The conditional probabilities relating various limbs and time instances are learned from training data, and particle filtering is used to perform the final pose inference.

Adaptive shape modeling. Another essential component of whole body modeling and tracking is the fitting of parameterized shape models to visual data. As we saw in Section 12.6.3 (Figure 12.19), the availability of large numbers of registered 3D range scans can be used to create *morphable models* of shape and appearance (Allen, Curless, and Popović 2003). Building on this work, Anguelov, Srinivasan, Koller *et al.* (2005) develop a sophisticated system called SCAPE (Shape Completion and Animation for PEople), which first acquires a large number of range scans of different people and of one person in different poses, and then registers these scans using semi-automated marker placement. The registered datasets are used to model the variation in shape as a function of personal characteristics and skeletal pose, e.g., the bulging of muscles as certain joints are flexed (Figure 12.21, top row). The resulting system can then be used for *shape completion*, i.e., the recovery of a full 3D mesh model from a small number of captured markers, by finding the best model parameters in both shape and pose space that fit the measured data.

Because it is constructed completely from scans of people in close-fitting clothing and uses a parametric shape model, the SCAPE system cannot cope with people wearing loose-fitting clothing. Bălan and Black (2008) overcome this limitation by estimating the body shape that fits within the visual hull of the same person observed in multiple poses, while Vlasic, Baran, Matusik *et al.* (2008) adapt an initial surface mesh fitted with a parametric shape model to better match the visual hull.

While the preceding body fitting and pose estimation systems use multiple views to estimate body shape, even more recent work by Guan, Weiss, Bălan *et al.* (2009) can fit a human shape and pose model to a single image of a person on a natural background. Manual initialization is used to estimate a rough pose (skeleton) and height model, and this is then used to segment the person's outline using the Grab Cut segmentation algorithm (Section 5.5). The shape and pose estimate are then refined using a combination of silhouette edge cues and shading information (Figure 12.21). The resulting 3D model can be used to create novel animations.

Activity recognition. The final widely studied topic in human modeling is motion, activity, and action recognition (Bobick 1997; Hu, Tan, Wang *et al.* 2004; Hilton, Fua, and Ronfard 2006). Examples of actions that are commonly recognized include walking and running, jumping, dancing, picking up objects, sitting down and standing up, and waving. Recent representative papers on these topics have been written by Robertson and Reid (2006), Sminchisescu, Kanaujia, and Metaxas (2006), Weinland, Ronfard, and Boyer (2006), Yilmaz and Shah (2006), and Gorelick, Blank, Shechtman *et al.* (2007).

12.7 Recovering texture maps and albedos

After a 3D model of an object or person has been acquired, the final step in modeling is usually to recover a *texture map* to describe the object's surface appearance. This first requires establishing a parameterization for the (u, v) texture coordinates as a function of 3D surface position. One simple way to do this is to associate a separate texture map with each triangle (or pair of triangles). More space-efficient techniques involve unwrapping the surface onto one or more maps, e.g., using a subdivision mesh (Section 12.3.2) (Eck, DeRose, Duchamp *et al.* 1995) or a geometry image (Section 12.3.3) (Gu, Gortler, and Hoppe 2002).

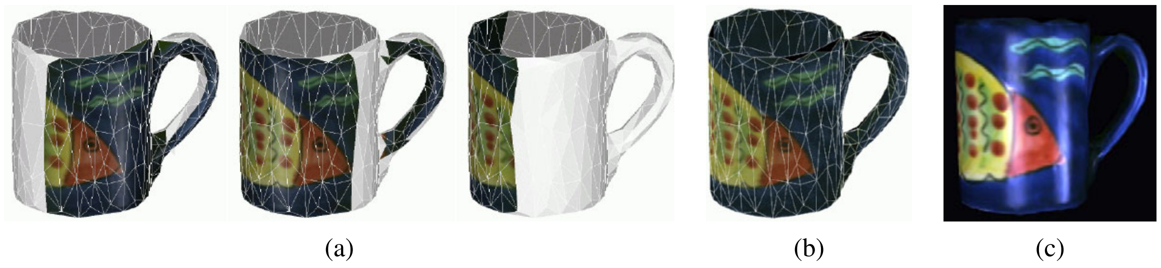


Figure 12.22 Estimating the diffuse albedo and reflectance parameters for a scanned 3D model (Sato, Wheeler, and Ikeuchi 1997) © 1997 ACM: (a) set of input images projected onto the model; (b) the complete diffuse reflection (albedo) model; (c) rendering from the reflectance model including the specular component.

Once the (u, v) coordinates for each triangle have been fixed, the perspective projection equations mapping from texture (u, v) to an image j's pixel (u_j, v_j) coordinates can be obtained by concatenating the affine $(u, v) \rightarrow (X, Y, Z)$ mapping with the perspective homography $(X, Y, Z) \rightarrow (u_j, v_j)$ (Szeliski and Shum 1997). The color values for the (u, v) texture map can then be re-sampled and stored, or the original image can itself be used as the texture source using projective texture mapping (OpenGL-ARB 1997).

The situation becomes more involved when more than one source image is available for appearance recovery, which is the usual case. One possibility is to use a *view-dependent texture map* (Section 13.1.1), in which a different source image (or combination of source images) is used for each polygonal face based on the angles between the virtual camera, the surface normals, and the source images (Debevec, Taylor, and Malik 1996; Pighin, Hecker, Lischinski *et al.* 1998). An alternative approach is to estimate a complete Surface Light Field for each surface point (Wood, Azuma, Aldinger *et al.* 2000), as described in Section 13.3.2.

In some situations, e.g., when using models in traditional 3D games, it is preferable to merge all of the source images into a single coherent texture map during pre-processing. Ideally, each surface triangle should select the source image where it is seen most directly (perpendicular to its normal) and at the resolution best matching the texture map resolution.[14] This can be posed as a graph cut optimization problem, where the smoothness term encourages adjacent triangles to use similar source images, followed by blending to compensate for exposure differences (Lempitsky and Ivanov 2007; Sinha, Steedly, Szeliski *et al.* 2008). Even better results can be obtained by explicitly modeling geometric and photometric misalignments between the source images (Shum and Szeliski 2000; Gal, Wexler, Ofek *et al.* 2010).

These kinds of approaches produce good results when the lighting stays fixed with respect to the object, i.e., when the camera moves around the object or space. When the lighting is strongly directional, however, and the object is being moved relative to this lighting, strong shading effects or specularities may be present, which will interfere with the reliable recovery of a texture (albedo) map. In this case, it is preferable to explicitly undo the shading effects (Section 12.1) by modeling the light source directions and estimating the surface reflectance properties while recovering the texture map (Sato and Ikeuchi 1996; Sato, Wheeler, and Ikeuchi 1997; Yu and Malik 1998; Yu, Debevec, Malik *et al.* 1999). Figure 12.22 shows

[14] When surfaces are seen at oblique viewing angles, it may be necessary to blend different images together to obtain the best resolution (Wang, Kang, Szeliski *et al.* 2001).

the results of one such approach, where the specularities are first removed while estimating the matte reflectance component (albedo) and then later re-introduced by estimating the specular component k_s in a Torrance–Sparrow reflection model (2.91).

12.7.1 Estimating BRDFs

A more ambitious approach to the problem of view-dependent appearance modeling is to estimate a general bidirectional reflectance distribution function (BRDF) for each point on an object's surface. Dana, van Ginneken, Nayar *et al.* (1999), Jensen, Marschner, Levoy *et al.* (2001), and Lensch, Kautz, Goesele *et al.* (2003) present different techniques for estimating such functions, while Dorsey, Rushmeier, and Sillion (2007) and Weyrich, Lawrence, Lensch *et al.* (2008) present more recent surveys of the topics of BRDF modeling, recovery, and rendering.

As we saw in Section 2.2.2 (2.81), the BRDF can be written as

$$f_r(\theta_i, \phi_i, \theta_r, \phi_r; \lambda), \tag{12.9}$$

where (θ_i, ϕ_i) and (θ_r, ϕ_r) are the angles the incident $\hat{\boldsymbol{v}}_i$ and reflected $\hat{\boldsymbol{v}}_r$ light ray directions make with the local surface coordinate frame $(\hat{\boldsymbol{d}}_x, \hat{\boldsymbol{d}}_y, \hat{\boldsymbol{n}})$ shown in Figure 2.15. When modeling the appearance of an object, as opposed to the appearance of a patch of material, we need to estimate this function at every point (x, y) on the object's surface, which gives us the *spatially varying* BRDF, or SVBRDF (Weyrich, Lawrence, Lensch *et al.* 2008),

$$f_v(x, y, \theta_i, \phi_i, \theta_r, \phi_r; \lambda). \tag{12.10}$$

If sub-surface scattering effects are being modeled, such as the long-range transmission of light through materials such as alabaster, the eight-dimensional bidirectional scattering-surface reflectance-distribution function (BSSRDF) is used instead,

$$f_e(x_i, y_i, \theta_i, \phi_i, x_e, y_e, \theta_e, \phi_e; \lambda), \tag{12.11}$$

where the e subscript now represents the *emitted* rather than the *reflected* light directions.

Weyrich, Lawrence, Lensch *et al.* (2008) provide a nice survey of these and related topics, including basic photometry, BRDF models, traditional BRDF acquisition using *gonio reflectometry* (the precise measurement of visual angles and reflectances), multiplexed illumination (Schechner, Nayar, and Belhumeur 2009), skin modeling (Debevec, Hawkins, Tchou *et al.* 2000; Weyrich, Matusik, Pfister *et al.* 2006), and image-based acquisition techniques, which simultaneously recover an object's 3D shape and reflectometry from multiple photographs.

A nice example of this latter approach is the system developed by Lensch, Kautz, Goesele *et al.* (2003), who estimate locally varying BRDFs and refine their shape models using local estimates of surface normals. To build up their models, they first associate a *lumitexels*, which contains a 3D position, a surface normal, and a set of sparse radiance samples, with each surface point. Next, they cluster such lumitexels into materials that share common properties, using a Lafortune reflectance model (Lafortune, Foo, Torrance *et al.* 1997) and a divisive clustering approach (Figure 12.23a). Finally, in order to model detailed spatially varying appearance, each lumitexel (surface point) is projected onto the basis of clustered appearance models (Figure 12.23b).

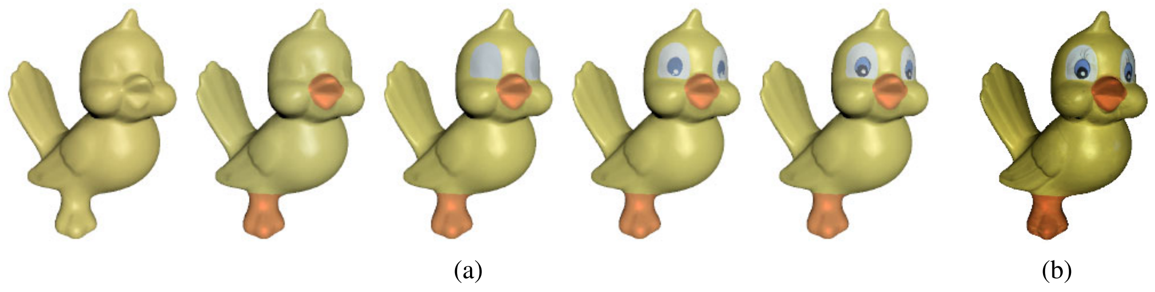


Figure 12.23 Image-based reconstruction of appearance and detailed geometry (Lensch, Kautz, Goesele *et al.* 2003) © 2003 ACM. (a) Appearance models (BRDFs) are re-estimated using divisive clustering. (b) In order to model detailed spatially varying appearance, each lumitexel is projected onto the basis formed by the clustered materials.

While most of the techniques discussed in this section require large numbers of views to estimate surface properties, a challenging future direction will be to take these techniques out of the lab and into the real world, and to combine them with regular and Internet photo image-based modeling approaches.

12.7.2 *Application*: 3D photography

The techniques described in this chapter for building complete 3D models from multiple images and then recovering their surface appearance have opened up a whole new range of applications that often go under the name *3D photography*. Pollefeys and Van Gool (2002) provide a nice introduction to this field, including the processing steps of feature matching, structure from motion recovery,[15] dense depth map estimation, 3D model building, and texture map recovery. A complete Web-based system for automatically performing all of these tasks, called ARC3D, is described by Vergauwen and Van Gool (2006) and Moons, Van Gool, and Vergauwen (2010). The latter paper provides not only an in-depth survey of this whole field but also a detailed description of their complete end-to-end system.

An alternative to such fully automated systems is to put the user in the loop in what is sometimes called *interactive computer vision*. van den Hengel, Dick, Thormhlen *et al.* (2007) describe their VideoTrace system, which performs automated point tracking and 3D structure recovery from video and then lets the user draw triangles and surfaces on top of the resulting point cloud, as well as interactively adjusting the locations of model vertices. Sinha, Steedly, Szeliski *et al.* (2008) describe a related system that uses matched vanishing points in multiple images (Figure 4.45) to infer 3D line orientations and plane normals. These are then used to guide the user drawing axis-aligned planes, which are automatically fitted to the recovered 3D point cloud. Fully automated variants on these ideas are described by Zebedin, Bauer, Karner *et al.* (2008), Furukawa, Curless, Seitz *et al.* (2009a), Furukawa, Curless, Seitz *et al.* (2009b), Mičušík and Košecká (2009), and Sinha, Steedly, and Szeliski (2009).

As the sophistication and reliability of these techniques continues to improve, we can expect to see even more user-friendly applications for photorealistic 3D modeling from images (Exercise 12.8).

[15] These earlier steps are also discussed in Section 7.4.4.

12.8 Additional reading

Shape from shading is one of the classic problems in computer vision (Horn 1975). Some representative papers in this area include those by Horn (1977), Ikeuchi and Horn (1981), Pentland (1984), Horn and Brooks (1986), Horn (1990), Szeliski (1991a), Mancini and Wolff (1992), Dupuis and Oliensis (1994), and Fua and Leclerc (1995). The collection of papers edited by Horn and Brooks (1989) is a great source of information on this topic, especially the chapter on variational approaches. The survey by Zhang, Tsai, Cryer *et al.* (1999) not only reviews more recent techniques but also provides some comparative results.

Woodham (1981) wrote the seminal paper of photometric stereo. Shape from texture techniques include those by Witkin (1981), Ikeuchi (1981), Blostein and Ahuja (1987), Garding (1992), Malik and Rosenholtz (1997), Liu, Collins, and Tsin (2004), Liu, Lin, and Hays (2004), Hays, Leordeanu, Efros *et al.* (2006), Lin, Hays, Wu *et al.* (2006), Lobay and Forsyth (2006), White and Forsyth (2006), White, Crane, and Forsyth (2007), and Park, Brocklehurst, Collins *et al.* (2009). Good papers and books on depth from defocus have been written by Pentland (1987), Nayar and Nakagawa (1994), Nayar, Watanabe, and Noguchi (1996), Watanabe and Nayar (1998), Chaudhuri and Rajagopalan (1999), and Favaro and Soatto (2006). Additional techniques for recovering shape from various kinds of illumination effects, including inter-reflections (Nayar, Ikeuchi, and Kanade 1991), are discussed in the book on shape recovery edited by Wolff, Shafer, and Healey (1992b).

Active rangefinding systems, which use laser or natural light illumination projected into the scene, have been described by Besl (1989), Rioux and Bird (1993), Kang, Webb, Zitnick *et al.* (1995), Curless and Levoy (1995), Curless and Levoy (1996), Proesmans, Van Gool, and Defoort (1998), Bouguet and Perona (1999), Curless (1999), Hebert (2000), Iddan and Yahav (2001), Goesele, Fuchs, and Seidel (2003), Scharstein and Szeliski (2003), Davis, Ramamoorthi, and Rusinkiewicz (2003), Zhang, Curless, and Seitz (2003), Zhang, Snavely, Curless *et al.* (2004), and Moons, Van Gool, and Vergauwen (2010). Individual range scans can be aligned using 3D correspondence and distance optimization techniques such as *iterated closest points* and its variants (Besl and McKay 1992; Zhang 1994; Szeliski and Lavallée 1996; Johnson and Kang 1997; Gold, Rangarajan, Lu *et al.* 1998; Johnson and Hebert 1999; Pulli 1999; David, DeMenthon, Duraiswami *et al.* 2004; Li and Hartley 2007; Enqvist, Josephson, and Kahl 2009). Once they have been aligned, range scans can be merged using techniques that model the signed distance of surfaces to volumetric sample points (Hoppe, DeRose, Duchamp *et al.* 1992; Curless and Levoy 1996; Hilton, Stoddart, Illingworth *et al.* 1996; Wheeler, Sato, and Ikeuchi 1998; Kazhdan, Bolitho, and Hoppe 2006; Lempitsky and Boykov 2007; Zach, Pock, and Bischof 2007b; Zach 2008).

Once constructed, 3D surfaces can be modeled and manipulated using a variety of three-dimensional representations, which include triangle meshes (Eck, DeRose, Duchamp *et al.* 1995; Hoppe 1996), splines (Farin 1992, 1996; Lee, Wolberg, and Shin 1996), subdivision surfaces (Stollnitz, DeRose, and Salesin 1996; Zorin, Schröder, and Sweldens 1996; Warren and Weimer 2001; Peters and Reif 2008), and geometry images (Gu, Gortler, and Hoppe 2002). Alternatively, they can be represented as collections of point samples with local orientation estimates (Hoppe, DeRose, Duchamp *et al.* 1992; Szeliski and Tonnesen 1992; Turk and O'Brien 2002; Pfister, Zwicker, van Baar *et al.* 2000; Alexa, Behr, Cohen-Or *et al.* 2003; Pauly, Keiser, Kobbelt *et al.* 2003; Diebel, Thrun, and Brünig 2006; Guennebaud and Gross

2007; Guennebaud, Germann, and Gross 2008; Oztireli, Guennebaud, and Gross 2008). They can also be modeled using implicit inside–outside characteristic or signed distance functions sampled on regular or irregular (octree) volumetric grids (Lavallée and Szeliski 1995; Szeliski and Lavallée 1996; Frisken, Perry, Rockwood *et al.* 2000; Dinh, Turk, and Slabaugh 2002; Kazhdan, Bolitho, and Hoppe 2006; Lempitsky and Boykov 2007; Zach, Pock, and Bischof 2007b; Zach 2008).

The literature on model-based 3D reconstruction is extensive. For modeling architecture and urban scenes, both interactive and fully automated systems have been developed. A special journal issue devoted to the reconstruction of large-scale 3D scenes (Zhu and Kanade 2008) is a good source of references and Robertson and Cipolla (2009) give a nice description of a complete system. Lots of additional references can be found in Section 12.6.1.

Face and whole body modeling and tracking is a very active sub-field of computer vision, with its own conferences and workshops, e.g., the International Conference on Automatic Face and Gesture Recognition (FG), the IEEE Workshop on Analysis and Modeling of Faces and Gestures, and the International Workshop on Tracking Humans for the Evaluation of their Motion in Image Sequences (THEMIS). Recent survey articles on the topic of whole body modeling and tracking include those by Forsyth, Arikan, Ikemoto *et al.* (2006), Moeslund, Hilton, and Krüger (2006), and Sigal, Balan, and Black (2010).

12.9 Exercises

Ex 12.1: Shape from focus Grab a series of focused images with a digital SLR set to manual focus (or get one that allows for programmatic focus control) and recover the depth of an object.

1. Take some calibration images, e.g., of a checkerboard, so you can compute a mapping between the amount of defocus and the focus setting.

2. Try both a fronto-parallel planar target and one which is slanted so that it covers the working range of the sensor. Which one works better?

3. Now put a real object in the scene and perform a similar focus sweep.

4. For each pixel, compute the local sharpness and fit a parabolic curve over focus settings to find the most in-focus setting.

5. Map these focus settings to depth and compare your result to ground truth. If you are using a known simple object, such as sphere or cylinder (a ball or a soda can), it's easy to measure its true shape.

6. (Optional) See if you can recover the depth map from just two or three focus settings.

7. (Optional) Use an LCD projector to project artificial texture onto the scene. Use a pair of cameras to compare the accuracy of your shape from focus and shape from stereo techniques.

8. (Optional) Create an all-in-focus image using the technique of Agarwala, Dontcheva, Agrawala *et al.* (2004).

Ex 12.2: Shadow striping Implement the handheld shadow striping system of Bouguet and Perona (1999). The basic steps include the following.

1. Set up two background planes behind the object of interest and calculate their orientation relative to the viewer, e.g., with fiducial marks.

2. Cast a moving shadow with a stick across the scene; record the video or capture the data with a webcam.

3. Estimate each light plane equation from the projections of the cast shadow against the two backgrounds.

4. Triangulate to the remaining points on each curve to get a 3D stripe and display the stripes using a 3D graphics engine.

5. (Optional) remove the requirement for a known second (vertical) plane and infer its location (or that of the light source) using the techniques described by Bouguet and Perona (1999). The techniques from Exercise 10.9 may also be helpful here.

Ex 12.3: Range data registration Register two or more 3D datasets using either iterated closest points (ICP) (Besl and McKay 1992; Zhang 1994; Gold, Rangarajan, Lu *et al.* 1998) or octree signed distance fields (Szeliski and Lavallée 1996) (Section 12.2.1).

Apply your technique to narrow-baseline stereo pairs, e.g., obtained by moving a camera around an object, using structure from motion to recover the camera poses, and using a standard stereo matching algorithm.

Ex 12.4: Range data merging Merge the datasets that you registered in the previous exercise using signed distance fields (Curless and Levoy 1996; Hilton, Stoddart, Illingworth *et al.* 1996). You can optionally use an octree to represent and compress this field if you already implemented it in the previous registration step.

Extract a meshed surface model from the signed distance field using marching cubes and display the resulting model.

Ex 12.5: Surface simplification Use progressive meshes (Hoppe 1996) or some other technique from Section 12.3.2 to create a hierarchical simplification of your surface model.

Ex 12.6: Architectural modeler Build a 3D interior or exterior model of some architectural structure, such as your house, from a series of handheld wide-angle photographs.

1. Extract lines and vanishing points (Exercises 4.11–4.15) to estimate the dominant directions in each image.

2. Use structure from motion to recover all of the camera poses and match up the vanishing points.

3. Let the user sketch the locations of the walls by drawing lines corresponding to wall bottoms, tops, and horizontal extents onto the images (Sinha, Steedly, Szeliski *et al.* 2008)—see also Exercise 6.9. Do something similar for openings (doors and windows) and simple furniture (tables and countertops).

4. Convert the resulting polygonal meshes into a 3D model (e.g., VRML) and optionally texture-map these surfaces from the images.

Ex 12.7: Body tracker Download the video sequences from the HumanEva Web site.[16] Either implement a human motion tracker from scratch or extend the code on that Web site (Sigal, Balan, and Black 2010) in some interesting way.

Ex 12.8: 3D photography Combine all of your previously developed techniques to produce a system that takes a series of photographs or a video and constructs a photorealistic texture-mapped 3D model.

[16] http://vision.cs.brown.edu/humaneva/.