The 3D Model Acquisition Pipeline

Fausto Bernardini and Holly Rushmeier

IBM Thomas J. Watson Research Center, Yorktown Heights, New York, USA

Abstract

Three-dimensional (3D) image acquisition systems are rapidly becoming more affordable, especially systems based on commodity electronic cameras. At the same time, personal computers with graphics hardware capable of displaying complex 3D models are also becoming inexpensive enough to be available to a large population. As a result, there is potentially an opportunity to consider new virtual reality applications as diverse as cultural heritage and retail sales that will allow people to view realistic 3D objects on home computers.

Although there are many physical techniques for acquiring 3D data—including laser scanners, structured light and time-of-flight—there is a basic pipeline of operations for taking the acquired data and producing a usable numerical model. We look at the fundamental problems of range image registration, line-of-sight errors, mesh integration, surface detail and color, and texture mapping. In the area of registration we consider both the problems of finding an initial global alignment using manual and automatic means, and refining this alignment with variations of the Iterative Closest Point methods. To account for scanner line-of-sight errors we compare several averaging approaches. In the area of mesh integration, that is finding a single mesh joining the data from all scans, we compare various methods for computing interpolating and approximating surfaces. We then look at various ways in which surface properties such as color (more properly, spectral reflectance) can be extracted from acquired imagery. Finally, we examine techniques for producing a final model representation that can be efficiently rendered using graphics hardware.

Keywords: 3D scanning, range images, reflectance models, mesh generation, texture maps sensor fusion

ACM CSS: I.2.10 Vision and Scene Understanding—*Modeling and recovery of physical attributes, shape, texture*; I.3.5 Computational Geometry and Object Modeling—*Geometric algorithms, languages and systems*; I.3.7 Three-Dimensional Graphics and Realism—*Color, shading, shadowing, and texture*; I.4.1 Digitization and Image Capture—*Reflectance, sampling, scanning*

1. Introduction

The past few years have seen dramatic decreases in the cost of three-dimensional (3D) scanning equipment, as well as in the cost of commodity computers with hardware graphics display capability. These trends, coupled with increasing Internet bandwidth, are making the use of complex 3D models accessible to a much larger audience. The potential exists to expand the use of 3D models beyond the well established games market to new applications ranging from virtual museums to e-commerce. To realize this potential, the pipeline from data capture to usable 3D model must be further developed. In this report we examine the state of the art of the processing of the output of range scanners into efficient numerical representations of objects for computer graphics applications.

Three-dimensional scanning has been widely used for many years for reverse engineering and part inspection [1]. Here we focus on acquiring 3D models for computer graphics applications. By 3D model, we refer to a numerical description of an object that can be used to render images of the object from arbitrary viewpoints and under arbitrary lighting conditions. We consider models that can be used to simulate the appearance of an object in novel synthetic environments. Furthermore, the models should be editable to provide the capability of using existing physical objects as the starting point for the design of new objects in computer modeling systems. The geometry should be editable—i.e. holes can be cut, the object can be stretched, or appended to other objects. The surface appearance properties should also be editable—i.e. surfaces can be changed from shiny to dull, or the colors of the surface can be changed.

To achieve this flexibility in the use of scanned objects, we consider systems which output shape in the form of clouds of points that can be connected to form triangle meshes, and/or fitted with NURBS or subdivision surfaces. The 3D points are augmented by additional data to specify surface finish and color. With the exception of surfaces with relatively uniform spatial properties, fine scale surface properties such as finish and color are ultimately stored as image maps covering the geometry.

The shape of 3D objects may be acquired by a variety of techniques, with a wide range in the cost of the acquisition hardware and in the accuracy and detail of the geometry obtained. On the high cost end, an object can be CAT scanned [2], and a detailed object surface can be obtained with isosurface extraction techniques. On the low cost end, models with relatively sparse 3D spatial sampling can be constructed from simple passive systems such as video streams by exploiting structure from motion [3], or by observing silhouettes and using space carving techniques [4].

In this report we focus on scanning systems that capture range images—that is an array of depth values for points on the object from a particular viewpoint. While these scanners span a wide range of cost, they are generally less expensive and more flexible than full 3D imaging systems such as CAT scanners, while obtaining much more densely sampled shapes than completely passive systems. We briefly review various types of range image scanners, and the principles they work on. However, for this report we consider a range scanner as a generic component, and consider the model building process given range images as input.

The process of building models from a range scanning system is shown in Figure 1. There are fundamentally two streams of processing—one for the geometry, and one for the fine scale surface appearance properties. As indicated by the dotted lines, geometric and surface appearance information can be exchanged between the two processing streams to improve both the quality and efficiency of the processing of each type of data. In the end, the geometry and fine scale surface appearance properties are combined into a single compact numerical description of the object.

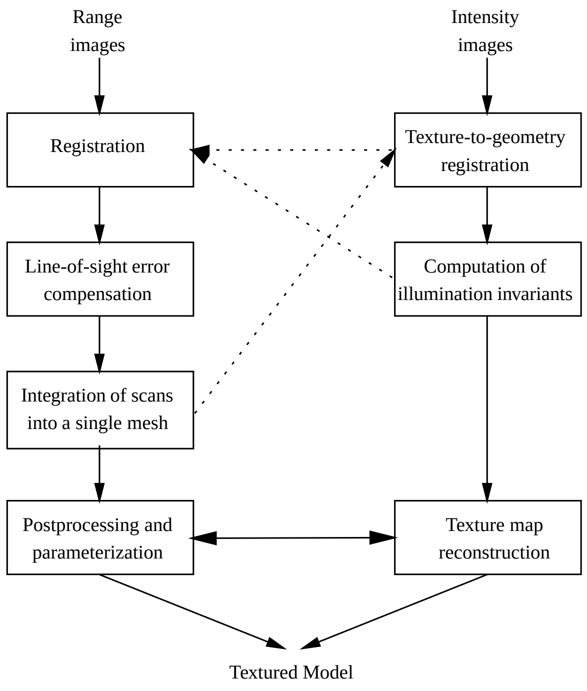
Range images
Intensity images
Registration
Texture-to-geometry registration
Line-of-sight error compensation
Computation of illumination invariants
Integration of scans into a single mesh
Postprocessing and parameterization
Texture map reconstruction
Textured Model

Figure 1: *The sequence of steps required for the reconstruction of a model from multiple overlapping scans.*

2. Scanning Hardware

Many different devices are commercially available to obtain range images. Extensive lists of vendors are maintained at various web sites. To build a model, a range scanner can be treated as a “black box” that produces a cloud of 3D points. It is useful however to understand the basic physical principles used in scanners. Characteristics of the scanner should be exploited to generate models accurately and efficiently.

The most common range scanners are triangulation systems, shown generically in Figure 2. A lighting system projects a pattern of light onto the object to be scanned—possibly a spot or line produced by a laser, or a detailed pattern formed by an ordinary light source passing through a mask or slide. A sensor, frequently a CCD camera, senses the reflected light from the object. Software provided with the scanner computes an array of depth values, which can be converted to 3D point positions in the scanner coordinate systems, using the calibrated position and orientation of the light source and sensor. The depth calculation may be made robust by the use of novel optics, such as the laser scanning systems developed at the National Research Council of Canada [5]. Alternatively, calculations may be made robust by using multiple sensors [6]. A fundamental limitation of what can be scanned with a triangulation system is having an adequate clear view for both the source and sensor to see the surface point currently being scanned. Surface reflectance properties affect the quality of data that can be obtained. Triangulation scanners may perform poorly on materials that are shiny, have low surface albedo, or that have significant subsurface scattering.

An alternative class of range scanners are time-of-flight systems. These systems send out a short pulse of light, and estimate distance by the time it takes the reflected light to return. These systems have been developed with near real time rates, and can be used over large (e.g. 100 m) distances. Time-of-flight systems require high precision in time measurements, and so errors in time measurement fundamentally limit how accurately depths are measured.

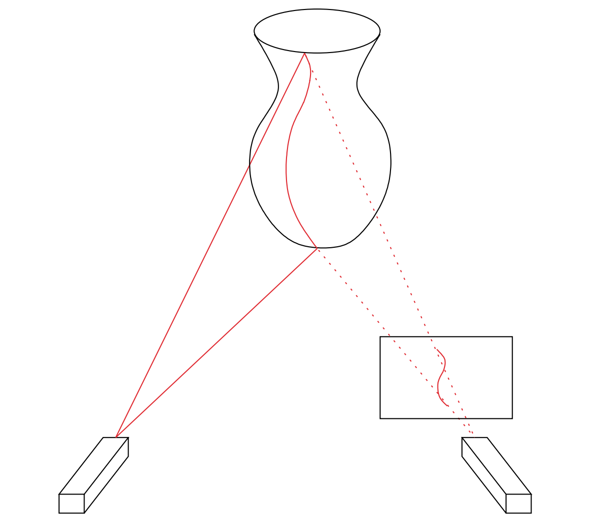

Figure 2: *Principles of a laser triangulation system. A laser projector shines a thin sheet of light onto the object. The CCD sensor detects, on each scan line, the peak of reflected laser light. 3D point positions are computed by intersecting the line through the pixel with the known plane of laser light.*

Basic characteristics to know about a range scanner are its scanning resolution, and its accuracy. Accuracy is a statement of how close the measured value is to the true value. The absolute accuracy of any given measurement is unknown, but a precision that is a value for the standard deviation that typifies the distribution of distances of the measured point to true point can be provided by the manufacturer. The tests used by manufacturers to determine precision are based on standard tests for length measurement developed for coordinate measurement machines or surveying applications, depending on the scale of the application. The absolute value of error increases with distance between the scanner and object. The deviation of measurements is a thin ellipsoid rather than a sphere—the error is greatest along the line-of-sight of the sensor. The precision of the measurements may vary across a range image. There are some effects that produce random errors of comparable magnitude at each point. Other effects may be systematic, increasing the error towards the edges of the scan. Because models are built from points acquired from many different range images, it is important to understand the relative reliability of each point to correctly combine them.

Resolution is the smallest distance between two points that the instrument measures. The accuracy of measured 3D points may be different than the resolution. For example, a system that projects stripes on an object may be able to find the depth at a particular point with submillimeter accuracy. However, because the stripes have some width, the device may only be able to acquire data for points spaced millimeters apart on the surface. Resolution provides a fundamental bound on the dimensions of the reconstructed surface elements, and dictates the construction of intermediate data structures used in forming the integrated representation.

Range scanners do not simply provide clouds of 3D points [7], but implicitly provide additional information. Simply knowing a ray from each 3D point to the scanning sensor indicates that there are no occluding surfaces along that ray, and provides an indicator of which side of the point is outside the object. Since range images are organized as two-dimensional (2D) arrays, an estimate of the surface normal at each point can be obtained by computing vector cross products for vectors from each point to its immediate neighbors. These indicators of orientation can be used to more efficiently reconstruct a full surface from multiple range images.

3. Registration

For all but the simplest objects, multiple range scans must be acquired to cover the whole object's surface. The individual range images must be aligned, or registered, into a common coordinate system so that they can be integrated into a single 3D model.

In high-end systems registration may be performed by accurate tracking. For instance, the scanner may be attached to a coordinate measurement machine that tracks its position and orientation with a high degree of accuracy. Passive mechanical arms as well as robots have been used. Optical tracking can also be used, both of features present in the scene or of special fiducial markers attached to the model or scanning area.

In less expensive systems an initial registration is found by scanning on a turntable, a simple solution that limits the size and geometric complexity of scanable objects (they must fit on the turntable and the system provides only a cylindrical scan which cannot re-construct self-occluding objects), and that leaves unsolved the problem of registration for scans of the top and bottom of the object. Many systems rely on interactive alignment: a human operator is shown side-by-side views of two overlapping scans, and must identify three or more matching feature points on the two images which are used to compute a rigid transformation that aligns the points.

Automatic feature matching for computing the initial alignments is an active area of research (recent work includes [8–12]). The most general formulation of the problem, that makes no assumptions on type of features (in the range and/or associated intensity images) and initial approximate registration is extremely hard to solve. Approximate position and orientation of the scanner can be tracked with fairly inexpensive hardware in most situations, and can be used as a starting point to avoid searching a large parameter space.

3.1. Registration of two views

Neither the controlled motion nor the feature matching techniques can usually achieve the same degree of accuracy as the range measurements. The initial alignment must therefore be refined by a different technique. The most successful approach to solve this problem has been the Iterative Closest Point (ICP) algorithm, originally proposed by Besl and McKay [13], Chen and Medioni [14], and Zhang [15].

The ICP algorithm consists of two steps: in the first step, pairs of candidate corresponding points are identified in the area of overlap of two range scans. Subsequently, an optimization procedure computes a rigid transformation that reduces the distance (in the least-squares sense) between the two sets of points. The process is iterated until some convergence criterion is satisfied. The general idea is that at each iteration the distance between the two scans is reduced, allowing for a better identification of true matching pairs, and therefore an increased chance of a better alignment at the next iteration. It has been proved [13] that the process converges to a local minimum, and in good implementations it does so in few steps. However, the algorithm may or may not converge to a global minimum, depending on the initial configuration. One obvious problem arises with surfaces that have few geometric features: two aligned partial scans of a cylindrical surface can slide relative to each other while the distance between corresponding points remains zero. When available, features in co-acquired texture images can help solve this underconstrained problems (see Section 3.3).

Variations of the algorithm differ in how the candidate matching pairs are identified, which pairs are used in computing the rigid transformation, and in the type of optimization procedure used. Besl and McKay [13] use the Euclidean closest point as the matching candidate to a given point. Chen and Medioni [14] find the intersection between a line normal to the first surface at the given point and the second surface, then minimize the distance between the given point and the tangent plane to the second surface at the intersection point. This technique has two advantages: it is less sensitive to non-uniform sampling, and poses no penalty for two smooth surfaces sliding tangentially one with respect to the other, a desirable behavior because in flat areas false matches can easily occur. See Figures 3 and 4.

Points from the first surface (*control* points) can be selected using uniform subsampling, or by identifying surface features. The set of candidate pairs can be weighted and/or pruned based on estimates of the likelihood of an actual match, and confidence in the data. Zhang [15] introduces a maximum tolerable distance and an orientation consistency check to filter out spurious pairings. Dorai *et al.* [16] model sensor noise and study the effect of measurement errors on the computation of surface normals. They employ a minimum variance estimator to formulate the

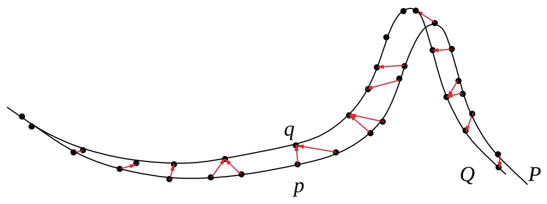

Figure 3: *One step of the ICP algorithm. Point matches are defined based on shortest Euclidean distance. Scan P is then transformed to minimize the length of the displacement vectors, in the least-squares sense.*

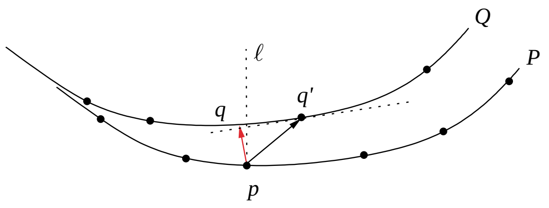

Figure 4: *In Chen and Medioni's method, a matching pair is created between a control point p on scan P and the closest point q on the tangent plane to Q at q'. q' is the sample point on Q closest to the intersection with the line ℓ perpendicular to P in p.*

error function to be minimized. They report more accurate registration results than Chen and Medioni's original method in controlled experiments. In related work, Dorai *et al.* [17] check distance constraints (given points p_1 and p_2 on the first surface, and corresponding points q_1, q_2 on the second surface, $|\|p_1 - p_2\| - \|q_1 - q_2\|| < \varepsilon$ must hold) to prune incompatible matches, also leading to improved registration results. Many researchers have proposed incorporating other features for validating matches: for example thresholding the maximum distance, discarding matches along surface discontinuities, evaluating visibility, and comparing surface normals, curvature or surface color information (see for example the good review in [18]). Use of the texture images as an aid to registration is further discussed in Section 3.3.

Given the two sets of matching points $P = \{p_1, \ldots, p_n\}$, $Q = \{q_1, \ldots, q_n\}$, the next problem is computing a rotation matrix R and translation vector T such that the sum of squares of pair wise distances

$$e = \sum_{i=1}^{n} \|p_i - (Rq_i + T)\|^2$$

is minimized. This problem can be solved in closed form by expressing the rotation as a quaternion [19], by linearizing the small rotations [14], or by using the Singular Value Decomposition. More statistically robust approaches have been investigated to avoid having to preprocess the data to eliminate outliers [20,21].

3.2. Registration of multiple views

When pair wise registration is used sequentially to align multiple views errors accumulate, and the global registration is far from optimal. Turk and Levoy [22] use a cylindrical scan that covers most of the surface of the object, and then incrementally register other scans to it. In their variation of ICP, they compute partial triangle meshes from the range scans, then consider the distance from each vertex of one mesh to the triangulated surface representing the other scan.

Bergevin *et al.* [23] extend the incremental approach to handle multiple views. One of the views is selected as the central (or reference) view. All the other views are transformed into the reference frame of the central view. At each iteration, each view is registered with respect to all other views using a varation of Chen and Medioni's method. The process is repeated until all incremental registration matrices are close to the identity matrix. Benjemaa and Schmitt [24] use a similar approach, but accelerate finding matching pairs by resampling the range images from a common direction of projection, and then performing the searches for the closest points on these images.

Pulli [25] describes another incremental multiview registration method that is particularly suited to the registration of large datasets. Pulli's method consists of two steps: in the first step, range scans are registered pair wise using Chen and Medioni's method. Matching points are discarded if they lie on scan's boundaries, if the estimated normals differ by more than a constant threshold, or when their distance is too large. A dynamic fraction, that increases as the registration gradually improves, of the best remaining pairs (the shorter ones) is then used for the alignment. After this initial registration, the overlap areas of each pair of scans is uniformly sampled, and the relative position of sample points stored and used in the successive step: the algorithm will assume that the pair wise registration is exact and will try to minimize relative motion. The second step considers the scans one at a time, and aligns each to the set of scans already considered. An inner loop in the algorithm considers all the scans that overlap with the current scan, and recursively aligns each of these scans until the relative change is smaller than a threshold, diffusing error evenly among all scans. By using a small number of pairs of points in the global registration phase, the need to have all the scans in memory is eliminated.

Blais and Levine [26] search for a simultaneous solution of all the rigid motions using a simulated annealing algorithm. Execution times for even just a few views are reportedly long. Neugebauer [27] uses the Levenberg–Marquardt method to solve a linearized version of the least-squares problem. A resolution hierarchy is used to improve robustness and efficiency. Invalid matches are detected and discarded at each iteration.

A different class of methods models the problem by imagining a set of springs attached to point pairs, and simulating the relaxation of the dynamic system. Stoddart and Hilton [28] assume that point pairs are given and remain fixed. Eggert *et al.* [18] link each data point to the corresponding tangent plane in another view with a spring. They use a hierarchical subsampling that employs an increasing number of control points as the algorithm progresses, and update correspondences at each iteration. They report better global registration error and a larger radius of convergence than other methods, at the expense of longer computation times. Their method also assumes that each portion of the object surface appears in at least two views.

3.3. Using the textures to aid registration

Images that record the ambient light reflected from an object (rather than a structured light pattern used for triangulation) may also be captured coincidently with the range images. Color or grayscale images are recorded to be used at texture maps (see Section 7). Range and texture images in systems that acquire both coincidently are registered to one another by calibration. That is, the relative position and orientation of the texture and range sensors are known, and so the projective mapping of the texture image onto the range image is known. When texture images registered to the range images are available, they may be used in the scan registration process. This is particularly advantageous when the texture images have a higher spatial resolution than the range images, and/or the object itself has features in the surface texture in areas that have few geometric features.

Texture images may be used in the initial alignment phase. Gagnon *et al.* [29] use texture data to assist a human operator in the initial alignment. Pairs of range images are aligned manually by marking three points on overlapping texture images. The locations of the matching points are refined by an algorithm that searches in their immediate neighborhoods using image cross-correlation [30]. A least-squares optimization follows to determine a general 3D transformation between the scans that minimizes the distances between the point pairs.

Roth [9] used textures in an automatic initial alignment procedure. "Interest" points in each texture image, such as corners, are identified using any of a variety of image processing techniques. A 3D Delaunay tetrahedralization is computed for all interest points in each scan. All matching triangles are found from pairs of potentially overlapping scans, and the transformation that successfully registers the most matching triangles is used. The advantage of using the triangles is that it imposes a rigidity constraint that helps insure that the matches found are valid. The method requires an adequate number of “interest” points in the textures. However, a relatively sparse pattern of points can be projected onto an object using laser light to guarantee that such points are available. Projected points were added to texture maps in the case study presented by Bernardini and Rushmeier [31], however the number of points per scan were not adequate for a completely automatic initial alignment.

Texture images may also be used in the refinement of the initial alignment. In general, there are two major approaches to using texture image data in the refinement phase. In one approach, the color image values are used as additional coordinates defining each point captured in the scan. In the other approach, matching operations are performed using the images directly.

Johnson and Kang [32,33] describe a method in which they use color from a texture as an additional coordinate for each point in an ICP optimization. Because the range images they use are of lower spatial resolution than the texture images, the range images are first supersampled to the texture resolution, and a color triplet is associated with each 3D point. The color triplets need to be adjusted to be comparable in influence to the spatial coordinates. They recommend scaling the color coordinates so that the range of values matches the range of values in the spatial coordinates. Further, to minimize image-to-image illumination variations they recommend using color in terms of YIQ rather than RGB, and applying a scale factor to the luminance, Y coordinate, that is much smaller than the chrominance IQ coordinates. The closest point search now becomes a search in 6D space, and a 6D k-d tree is used to accelerate the search. For tests using scanned models of rooms which have many planar areas with high texture variation, they demonstrate order of magnitude reductions in alignment errors. Schütz *et al.* [34] present a similar extended-coordinate ICP method, that uses scaled normals data (with normals derived from the range data) as well as color data.

The alternative approach to using texture image data is to perform matching operations on image data directly. This allows image structure to be exploited, and avoids search in high dimensional coordinate space. To compare texture images directly, these types of methods begin by using the range scan and an initial estimate of registration to project the texture images into a common view direction, as illustrated in Figure 5.

Weik [35] projects both the texture image and the texture gradient image of a source scan to be aligned with a second destination scan. The difference in intensities in the two images in the same view are then computed. The texture difference image and gradient image are then used to estimate the locations of corresponding points in the two images. A rigid transformation is then computed that minimizes the sum of the 3D distances between the corresponding point pairs. Pulli [36] describes a method similar to Weik’s that replaces the use of image gradient and differences with a full image registration to find corresponding points. Pulli’s technique uses a version of planar perspective warping described by Szeliski and Shum [37] for image registration. To make the registration more robust, Pulli describes a hierarchical implementation. Similar to Kang and Johnson, Pulli examines alternative color spaces to minimize the effects of illumination variations. For the test cases used—small objects with rich geometric and textural features—there appears to be no advantage of using images in color spaces other than RGB.

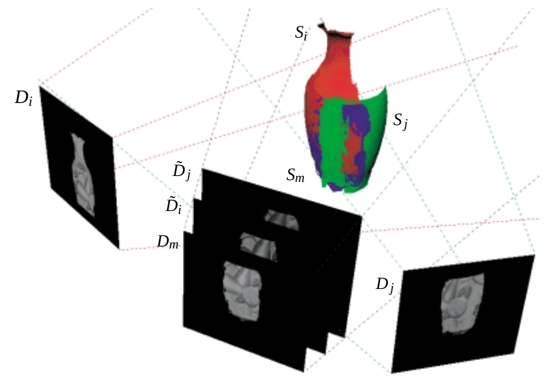

Figure 5: *Registration methods that work with images begin by projecting overlapping textures into the same view. Here geometries S_i and S_j are used to project the corresponding texture maps D_i and D_j into the same view as a third scan S_m.*

Both Weik’s and Pulli’s methods require operations on the full high-resolution texture images. A high degree of overlap is required, and scan-to-scan variability in illumination introduces error. Fine scale geometry is matched only if these details are revealed by lighting in the images. Both methods can be effective if there are substantial albedo variations in the scans that dominate illumination variations.

Bernardini *et al.* [38] present a registration method that combines elements of several of the other texture-based techniques. The initial alignment is first refined with a purely geometric ICP. Similar to Weik and Pulli, the texture images are projected into a common view. Similar to Roth, feature points are located in the texture images. However, unlike Roth the method does not attempt to match feature points. Rather, similar to the approach by Gagnon *et al.* the initial correspondences are refined by doing a search in a small neighborhood around each point, and finding corresponding pixels where an image cross-correlation measure is minimized. A rigid rotation is then found that minimizes the distance between the newly identified corresponding points.

3.4. Future directions

Successful refinement of an initial registration has been demonstrated for a large class of objects. This step does not appear to be a major obstacle to a fully automatic model-building pipeline. Robust solutions for the automatic alignment of totally uncalibrated views are not available, although some progress is being made. Scanner instrumentation with an approximate positioning device seems a feasible solution in most cases. Very promising is the use of improved feature-tracking algorithms from video sequences as an inexpensive way of producing the initial registration estimate.

4. Line-of-sight Error

After the scans have been aligned the individual points would ideally lie exactly on the surface of the reconstructed object. However, one still needs to account for residual error due to noise in the measurements, inaccuracy of sensor calibration, and imprecision in registration. The standard approach to deal with the residual error is to define new estimates of actual surface points by averaging samples from overlapping scans. Often the specific technique used is chosen to take advantage of the data structures used to integrate the multiple views into one surface. Because of this, details of the assumed error model and averaging method are often lost or overlooked by authors. We believe that this problem is important enough to deserve a separate discussion. In addition, line-of-sight error compensation, together with resampling and outlier filtering, is a necessary preprocessing step for interpolatory mesh integration methods.

Among the first to recognize the need for a mathematical model of scanner inaccuracies and noise were Hébert *et al.* [40], in the context of data segmentation and polynomial section fitting. Their error model incorporates the effects of viewing angle and distance, and is expressed as an uncertainty ellipsoid defined by a Gaussian distribution. Other sources of non-Gaussian error, such as shadows, surface specularities and depth discontinuities, which generally produce outliers, are not included in the model. For a typical triangulation scanner the error in estimating the x, y position of each sample is much smaller than the error in estimating the depth z. Therefore the ellipsoid is narrow with its longer axis aligned with the direction towards the sensor, see Figure 6. Building on the work of Hébert *et al.* [40], Rutishauser *et al.* [39] define an optimal reconstruction of a surface from two sets of estimates, in the sense of probability theory. However, they have to resort to some approximations in their actual computations. For a measured point on one scan, they find the best matching point (again, in the probabilistic sense) on the triangle defined by the three closest samples on the second scan. The optimal estimation of point location is then computed using the modified Kalman minimum-variance estimator.

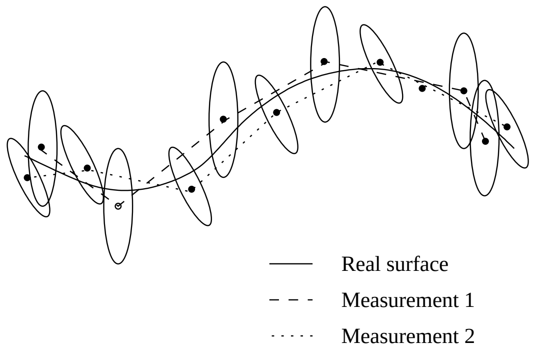

Figure 6: *Probabilistic model of measurement error (adapted from Rutishauser et al.* [39]*).*

Soucy and Laurendeau [41] model error in a laser triangulation system as proportional to the fraction of illuminance received by the sensor, expressed by the cosine square of the angle between the surface normal at the measured point and the sensor viewing direction. Overlapping range data is resampled on a common rectangular grid lying on a plane perpendicular to the average of the viewing directions of all contributing scans. Final depth values are computed as weighted averages of the resampled values, where the weight used is the same cosine square defined above. These points are then connected into a triangle mesh.

Turk and Levoy [22] employ a similar method, but invert the steps of creating a triangulated surface and finding better surface position estimates. In their approach individual range scans are first triangulated, then stitched together. In areas of overlap, vertices of the resulting mesh are moved along the surface normal to a position computed as the average of all the intersection of a line through the point in the direction of the normal and all the overlapping range scans.

Neugebauer [27] adjusts point positions along the scanner line-of-sight. He uses a weighted average where each weight is the product of three components: the first is the cosine of the angle between surface normal and sensor viewing direction (if the cosine is smaller than 0.1, the weight is set to zero); the second contribution is a function that approximates the square distance of a sample point to the scan boundary, allowing a smooth transition between scans; the third component is Tukey's biweight function, used to filter outliers. The weighting is applied iteratively.

In volumetric methods line-of-sight error compensation is done by computing a scalar field that approximates the signed distance to the true surface, based on a weighted average of distances from sample points on individual range scans. The details of the various methods will be discussed in the next section.

5. Scan Integration

For most applications, it is desirable to merge the aligned multiple scans into a unified, non-redundant surface representation. A significant amount of research in this direction has been done in the past. In this section, we will try to classify this work based on the type of assumptions and approach taken, and we will point to recent publications that are representative of each category, without trying to exhaustively cite the vast literature available on this subject. Previous reviews of work in this field include [42–44].

The goal of scan integration is to reconstruct the geometry and topology of the scanned object from the available data. The problem is difficult because in general the data points are noisy, they may contain outliers, parts of the surface may not have been reached by the scanner, and in general there is no guarantee that the sampling density is even sufficient for a correct reconstruction.

Some progress is being made in characterizing the problem more rigorously, at least in restricted settings. A first classification of methods can be made based on whether the input data is assumed to be unorganized points (*point cloud*) or a set of range scans. Techniques that deal with the first kind of input are more general, but also usually less robust in the presence of noise and outliers. The second category uses information in addition to simple point position, such as estimated surface normal, partial connectivity embedded in the range scan, sensor position, to better estimate the actual surface.

A second classification groups techniques based on the approach taken to reconstruct surface connectivity. A practical consequence of this choice is the *size* of the problem that can be solved using given computing resources. We will review selected work based on this second categorization.

5.1. Delaunay-based methods

The Delaunay complex $D(S)$ associated with a set of points S in R^3 decomposes the convex hull of S and imposes a connectivity structure. Delaunay-based methods reconstruct a surface by extracting a subcomplex from $D(S)$, a process sometime called *sculpting*. This class of algorithms usually assumes only a point cloud as input. A recent review and unified treatment of these methods appears in [45].

One technique to select an interesting subcomplex, in fact a parameterized family of subcomplexes, is based on alpha-shapes [46]. Bajaj *et al.* [44,47] use a binary search on the parameter α to find a subcomplex that defines a closed surface containing all the data points. Smaller concave features not captured by the alpha-shape are found with the use of heuristics. The surface is then used to define a signed distance. A C^1 implicit piecewise-polynomial function is then adaptively fit to the signed distance field.

A commercial software product by Geomagic is based on a different technique to extract the subcomplex, called the *wrap complex* [48]. The technique can handle non-uniform samplings, but requires some interactive input.

Amenta *et al.* [49,50] introduce the concept of *crust*, the subcomplex of the Delaunay complex of $S \cup P$, where P is the set of poles of the Voronoi cells of S, formed by only those simplices whose vertices belong to S. The poles of a sample point $s \in S$ are the two farthest vertices of its Voronoi cell. The algorithm automatically handles non-uniform samplings, and its correctness, under somewhat stringent sampling density conditions, has been proven, both in the sense of a topologically correct reconstruction and of convergence to the actual surface for increasing sampling density. Experimental results prove that the algorithm performs well in practice for much less dense samplings than the theoretical bound. Based on a similar concept, but leading to a more efficient and robust implementation is the *power crust* algorithm [51,52]. The first step of the power crust algorithm is to compute a piecewise-linear approximation of the *medial axis transform*, interpolating the poles P of the Voronoi cells of S, defined as above. The poles are weighted with the associate (approximate) radius of the maximal balls that do not intersect the surface. The second step computes a piecewise-linear approximation of the surface as a subset of the faces of the *power diagram* of the set of weighted poles. One additional benefit of the algorithm is that it produces a closed ("watertight") surface in the presence of uneven sampling density. Sampling assumptions and theoretical guarantees are defined in [52]. Practical extensions to deal with sharp features, holes and noise are discussed in [51]. Experimental results for datasets containing several hundred thousand points are shown.

Also using the concept of poles to define a local surface approximation is the *cocones* algorithm proposed by Amenta *et al.* [53]. Here the poles of the Voronoi diagram of P are used to define an approximate normal for each sample point. The complement (restricted to the Voronoi cell of the point) of a double cone centered at p, with axis aligned with the sample point normal and an aperture of $3\pi/8$, is defined as the *cocone* of p. It is proved that the cocone constitutes a good approximation for the surface in the neighborhood of p. The local surface reconstrucion is then defined by the collection of Delaunay triangles incident on p that are dual to Voronoi edges contained in the cocone of p. The union of all the local reconstruction constitutes a superset of the final manifold triangulation, which is obtained with a global *prune and walk* algorithm. These results are presented in the context of a practical implementation by Dey *et al.* [54]. The authors employ a divide and conquer method based on an octree partition of the input points to avoid a global Voronoi computation. The pointsets contained in each octree node are padded with enough points from neighboring nodes to enforce the computation of compatible triangulations along common boundaries. Again, a global prune and walk algorithms selects a manifold subset of the candidate triangles. The divide and conquer approach leads to reduced computation times and memory usage, allowing the treatment of datasets with millions of samples on common workstations.

In the context of Delaunay-based methods it is possible to study the sampling conditions the guarantee a correct reconstruction. Attempts so far have been mostly restricted to the 2D case [55–57], with the exception of [50] and [52]. The main shortcomings of these methods are their sensitivity to noise and outliers (these algorithms interpolate the data points, so outliers must be removed in preprocessing), and their computational complexity. Robustly computing and representing the connectivity of the 3D Delaunay complex can be a costly task. Experimental results are usually limited to "clean" datasets with less than a few hundred thousand points (with the exception of [54]).

5.2. Surface-based methods

Surface-based methods create the surface by locally parameterizing (or implicitly assuming a local parameterization of) the surface and connecting each points to its neighbors by local operations. Some methods make use of the partial connectivity implicit in the range images.

The *zippering* approach of Turk and Levoy [22] works by first individually triangulating all the range scans. The partial meshes are then eroded to remove redundant, overlapping triangles. The intersecting regions are then locally retriangulated and trimmed to create one seamless surface. Vertex positions are then readjusted to reduce error, as described in Section 4.

Soucy and Laurendeau [41] use canonical Venn diagrams to partition the data into regions that can be easily parameterized. Points in each region are resampled and averaged (see Section 4), and locally triangulated. Patches are then stitched together with a constrained Delaunay algorithm.

A recent paper by Bernardini *et al.* [58] describes an algorithm to interpolate a point cloud that is not based on sculpting a Delaunay triangulation. Their method follows a region growing approach, based on a *ball-pivoting* operation. A ball of fixed radius (approximately the spacing between two sample points) is placed in contact with three points, which form a seed triangle. The three edges initialize a queue of edges on the active boundary of the region. Iteratively, an edge is extracted from the queue, and the ball pivots around the extracted edge until it touches a new point. A new triangle is formed, the region boundary updated, and the process continues. The approach can easily be extended to restart with a larger ball radius to triangulate regions with sparser data points. This method was implemented to make efficient use of memory by loading at any time only the data in the region currently visited by the pivoting ball, rather than the entire dataset. This allowed the triangulation of a large collection of scans with millions of samples.

Gopi *et al.* [59] compute local 2D Delaunay triangulations by projecting each point and its neighborhood on a tangent plane, and then lift the triangulation to 3D.

Surface based methods can easily process large datasets, and can handle (and compensate for) small-scale noise in the data. Robustness issues arise when the noise makes it difficult to locally detect the correct topology of the surface.

5.3. Volumetric methods

Volumetric methods [60–62] are based on computing a signed distance field in a regular grid enclosing the data (usually, only in proximity of the surface), and then extracting the zero-set of the trivariate function using the marching cube algorithm [63]. The various approaches differ on the details of how the signed distance is estimated from the available data.

Curless and Levoy [60] compute the signed distance from each scan by casting a ray from the sensor through each voxel near the scan. The length of the ray from the voxel to the point in which it intersects the range surface is computed and accumulated at the voxel with values computed from other scans using weights dependent, as usual, on surface normal and viewing direction. This approach may lead to a biased estimate of surface location, as noted in [61]. Hilton *et al.* [62] also blend signed distances from individual scans, and use extra rules to handle correctly the case of of different surfaces in close proximity, both with the same and opposite orientation. Wheeler *et al.* [61] propose a solution that is less sensitive to noise, outliers, and orientation ambiguities. They assign to each voxel the signed distance to the closest point on the *consensus surface*, a weighted average of nearby measurements. Only measurements for which a user-specified quorum of samples with similar position and orientation is found are used.

Boissonnat and Cazals [64] use natural neighbor interpolation to define a global signed distance function. The natural neighbors of a point x are the neighbors of x in the Delaunay triangulation of $P \bigcup \{x\}$. Using natural neighbors avoids some of the pitfalls of other local surface approximations (for example taking just the points within a given distance from x, or its k closest neighbors). However, it requires the computation of a global Delaunay triangulation, which limits the size of the datasets that can be handled by the algorithm in practice. Since the Delaunay triangulation of the points must be computed, it can also be used as the starting point for the construction of piecewise-linear approximation of the surface that satisfies a user-specified tolerance. The initial approximation is formed by all those Delaunay triangles whose dual Voronoi edge is *bipolar*, that is such that the global signed distance function has different signs at its two endpoints. This triangulation is then incrementally refined until the tolerance condition is satisfied. Examples of reconstruction from datasets of moderate size are shown in the paper.

Volumetric methods are well suited for very large datasets. Once the individual range scans have been processed to accumulate signed distance values, storage and time complexity are output sensitive: they mainly depend on the chosen voxel size, or resolution of the output mesh. Memory usage can be reduced by explicitly representing only voxels in close proximity to the surface [60] and by processing the data in slices. The choice of voxel size is usually left to the user. Small voxels produce an unnecessarily large number of output triangles and increase usage of time and space. Large voxels lead to oversmoothing and loss of small features. These problems can be alleviated by using an adaptive sampling (e.g. octree rather than regular grid [65]) and/or by postprocessing the initial mesh with a data fitting procedure [66–68].

Volumetric methods are also well suited to producing water-tight models. By using the range images to carve out a spatial volume, an object definition can be obtained without holes in the surface. Reed and Allen [69] demonstrate the evolution of a solid model from a series of range images, with the data from each image carving away the solid that lies between the scanner and each sample point. Rocchini *et al.* [70] also describe a volumetric method that fills holes.

5.4. Deformable surfaces

Another class of algorithms is based on the idea of *deforming* an initial approximation of a shape, under the effect of external forces and internal reactions and constraints.

Terzopoulos *et al.* [71] use an elastically-deformable model with intrinsic forces that induce a preference for symmetric shapes, and apply them to the reconstruction of shapes from images. The algorithm is also capable of inferring non-rigid motion of an object from a sequence of images.

Pentland and Sclaroff [72] adopted an approach based on the finite element method and parametric surfaces. They start with a simple solid model (like a sphere or cylinder) and attach virtual “springs” between each data point and a point on the surface. The equilibrium condition of this dynamic system is the reconstructed shape. They also show how the set of parameters that describe the recovered shape can be used in object recognition.

Recently a number of methods based on the concept of levels sets have been proposed. These methods combine a robust statistical estimation of surface position in the presence of noise and outliers with an efficient framework for surface evolution. See e.g. [73,74].

6. Postprocessing

Postprocessing operations are often necessary to adapt the model resulting from scan integration to the application at hand. Very common is the use of mesh simplification techniques to reduce mesh complexity [75].

To relate a texture map to the integrated mesh, the surface must be parameterized with respect to a 2D coordinate system. A simple parameterization is to treat each triangle separately [32,76] and to pack all of the individual texture maps into a larger texture image. However, the use of mip-mapping in this case is limited since adjacent pixels in the texture may not correspond to adjacent points on the geometry. Another approach is to find patches of geometry which are height fields that can be parameterized by projecting the patch onto a plane. Stitching methods [2] use this approach by simply considering sections of the scanned height fields as patches.

Many parameterization methods have been developed for the general problem of texture mapping. Several methods seek to preserve the relative distance between 3D points in their pairing to a 2D coordinate system [77,78]. Marschner [79] describes an example of applying a relative distance preserving parameterization in a scanning application. The surface is subdivided into individual patches by starting with seed triangles distributed over the object, and growing regions around each seed. Harmonic maps are found to establish a 2D coordinate system for each patch, so individual patches need not be height fields.

Sloan *et al.* [80] have observed that maintaining relative distances may not produce optimal parameterizations for texture mapping. They suggest that uniform texture information, rather than distance preservation, should drive the parameterization. They applied this idea to synthetic textures only, but it may prove to be an effective approach in some scanning applications as well.

Another important step for applications that involve editing and animating the acquired model is the conversion of the mesh to a parametric, higher-order surface representation, for example using NURBS or a subdivision scheme.

The technique of Hoppe *et al.* [81] starts with a triangle mesh and produces a smooth surface based on Loop’s subdivision scheme [82]. Their method is based on minimizing an energy function that trades off conciseness and accuracy-of-fit to the data, and is capable of representing surfaces containing sharp features, such as creases and corners.

More recently, Eck and Hoppe [83] proposed an alternative surface fitting approach based on tensor-product B-spline patches. They start by using a signed-distance zero-surface extraction method [84]. An initial parameterization is built by projecting each data point onto the closest face. The method continues with building from the initial mesh a *base complex* (a quadrilateral-domain complex, with the same topology of the initial mesh) and a continuous parameterization from the base complex to the initial mesh, leveraging on the work of Eck *et al.* [78]. A tangent-plane continuous network of tensor-product B-spline patches, having the base complex as parametric domain, is then fit to the data points, based on the scheme of Peters [85]. The fitting process is cast as an iterative minimization of a functional, which is a weighted sum of the distance functional (the sum of square Euclidean distances of the data points from the surface) and a fairness functional (thin plate energy functional).

Another NURBS fitting technique is described by Krishnamurthy and Levoy [86]. The user interactively chooses how to partition the mesh into quadrilateral patches. Each polygonal patch is parametrized and resampled, using a spring model and a relaxation algorithm. Finally, a B-spline surface is fit to each quadrilateral patch. In addition, a displacement map is computed that captures the fine geometric detail present in the data.

Commercial packages that allow a semi-automated parametrization and fitting are available.

7. Texture

In addition to the overall shape of an object, the rendering of high quality images requires the fine scale surface appearance, which includes surface color and finish. We will refer to such properties generically as the surface texture. Beyond color and finish, texture may also include descriptions of fine scale surface geometry, such as high spatial-resolution maps of surface normals or bidirectional textures.

Surface color and finish are informal terms. Color is actually a perceived quantity, depending on the illumination of an object, human visual response, and the intrinsic spectral reflectance of the object. Finish—such as smoothness or gloss—is also not a directly acquired property, but is a consequence of an object's intrinsic reflectance properties. The fundamental quantity that encodes the intrinsic properties of the surface is the Bidirectional Reflectance Distribution Function (BRDF). To fully render an accurate image, the BRDF must be known for all points on a surface. The BRDF $f_r(\lambda, x, y, \boldsymbol{\omega}_i, \boldsymbol{\omega}_r)$ at a surface point (x, y) is the ratio of radiance reflected in a direction $\boldsymbol{\omega}_r$ to an incident energy flux density from direction $\boldsymbol{\omega}_i$ for wavelength λ. The BRDF can vary significantly with position, direction and wavelength. Most scanning systems consider detailed positional variations only, with wavelength variations represented by an RGB triplet, and Lambertian (i.e. uniform for all directions) behavior assumed. Furthermore, most scanning systems acquire relative estimates of reflectance, rather than attempting to acquire an absolute value.

Here we will consider how texture data is acquired, and then how it is processed to provide various types of BRDF estimates, and estimates of fine scale surface structure.

7.1. Texture-geometry registration

It is possible to capture the spectral reflectance of an object as points are acquired with a polychromatic laser scanner [87]. However, data for texture is typically acquired by an electronic color camera or using conventional color photographs that are subsequently scanned into electronic form. The texture images need to be registered with the acquired 3D points. The most straightforward system for doing this is registration by calibration. That is, color images corresponding to each range image are acquired at the same time, using a camera with a known, measured position and orientation relative to the sensor used for obtaining geometry. As discussed in Section 3.3, an advantage of this approach is that acquired texture can be used in the geometric registration process.

When textures are acquired separately from geometry, the texture-to-geometry registration is performed after the full mesh integration phase. Finding the camera position and orientation associated with a 2D image of a 3D object is the well-known camera calibration problem. Numerous references on solutions to this problem can be found in the Price's Computer Vision bibliography [88], Section 15.2, "Camera Calibration Techniques." Camera calibration involves estimating both the extrinsic and intrinsic parameters. The extrinsic parameters are the translation and rotation to place the camera viewpoint correctly in the object coordinate system. The intrinsic parameters include focal length and radial distortion. For objects which have an adequate number of unique geometric features, it is possible to manually identify pairs of corresponding points in the 2D images and on the numerical 3D object. Given such correspondences, classic methods such as that described by Tsai [89], can be used to register the captured color images to the 3D model [2].

For some objects it may not be possible for a user to find a large number of accurate 2D–3D correspondences. Neugebauer and Klein [90] describe a method for refining the registration of a group of existing texture images to an existing 3D geometric model. The method begins with a rough estimate of the camera parameters for each image in the set, based on correspondences that are not required to be highly accurate. The parameters for all of the texture images are improved simultaneously by assuming the intrinsic camera parameters are the same for all images, and enforcing criteria that attempt to match the object silhouettes in the image with the silhouette of the 3D model, and to match the image characteristics at locations in texture images that correspond to the same 3D point.

Lensch *et al.* [91] present a method for finding the camera position for texture images in terms of a geometric object coordinate system using comparisons of binary images. First, a binary version of each texture image is computed by segmenting the object from the background. This is compared to a synthetic binary image generated by projecting the known geometry into a camera view based on an initial guess of camera parameters. The values of the camera parameters are refined by using a downhill simplex method to minimize the difference between the binary texture image and the synthetic image. In a subsequent step the camera parameters for all views of an object are adjusted simultaneously to minimize the error in overlapping textures from neighboring views.

Nishino *et al.* [92] apply an alternative technique that relies on image intensities rather than identifying features or extracting contours. They employ the general approach developed by Viola [93] that formulates the alignment as the maximization of the mutual information between the 3D model and the texture image.

Rather than using an *ad hoc* method for deciding the positions for capturing texture images, Matsushita and Kaneko [94] use the existing 3D geometric model to plan the views for capturing texture. Methods to plan texture image capture can draw on the numerous computer vision techniques for view planning, e.g. see [88] Section 15.1.4.1, “Planning Sensor Position.” Matsushita and Kaneko develop a table of a set of candidate views, and the object facets that are visible in each view. Views are selected from the table to obtain the views that image the largest number of yet to be imaged facets. After the view set is selected, synthetic images which form the views are generated. For each synthetic image the real camera then is guided around the object to find the view that approximates the synthetic image, and a texture image is captured. The texture image to model registration is refined after capture using a variation of Besl and McKay’s ICP algorithm [13] that acts on points on the silhouettes of the real and synthetic images.

7.2. Illumination invariance

The goal of capturing texture is to obtain a surface description that is illumination invariant—i.e. intrinsic to the surface and independent of specific lighting conditions. The pixel values in an image acquired by an electronic camera depend on the environmental lighting and the camera transfer parameters as well as the object properties. Approximate illumination invariants can be obtained directly by appropriate lighting and camera design. More complete estimates require processing of the acquired images. The variety of techniques can be understood by examining the specific relationships between the physical acquisition equipment and the end numerical value stored in an image.

Figure 7 shows a generic simplified system for obtaining a texture image. A light source with radiance $L_s(\lambda, \boldsymbol{\omega}_s)$ in direction $\boldsymbol{\omega}_s$ from the normal of the source surface is at distance r_s from the object. Light incident from direction $\boldsymbol{\omega}_i$ is reflected with radiance $L_p(\lambda, \boldsymbol{\omega}_r)$ into the direction of

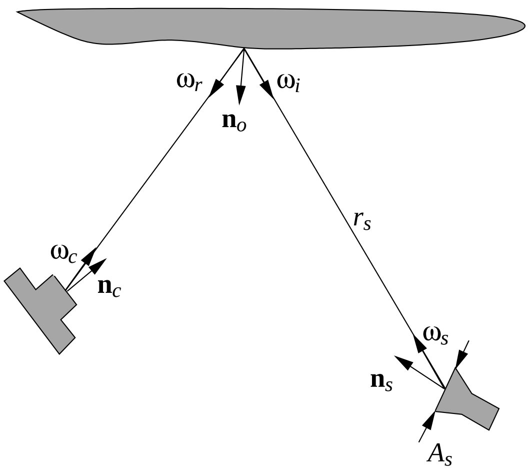

Figure 7: *Generic geometry of texture map acquisition.*

a pixel p. The radiance $L_p(\lambda)$ is related to the object BRDF by:

$$L_p(\lambda) = \int f_r(\lambda, x, y, \boldsymbol{\omega}_i, \boldsymbol{\omega}_r) \times L_s(\lambda, \boldsymbol{\omega}_s)\mathbf{n}_o \cdot \boldsymbol{\omega}_i \mathbf{n}_s \cdot \boldsymbol{\omega}_s \, \mathrm{d}A_s / r_s^2. \quad (1)$$

The energy per unit area and time $E_p(\lambda)$ incident on the pixel from direction $\boldsymbol{\omega}_c$ for an exposure time of τ is:

$$E_p(\lambda) = \tau \int L_p(\lambda)\mathbf{n}_c \cdot \boldsymbol{\omega}_c \, \mathrm{d}\Omega \quad (2)$$

where Ω is the solid angle of the object area viewed by the pixel, determined by the camera focal length and pixel size. This is converted to a 0–255 value (for an 8-bit sensor) C where C corresponds to the red (R), green (G) or blue (B) channel by:

$$C = K\left(\int_\lambda E_p(\lambda)s_C(\lambda)\,\mathrm{d}\lambda\right)^\gamma + C_o \quad (3)$$

where K is the system sensitivity, $s_C(\lambda)$ is the normalized sensor spectral response for channel C, C_o is the response for zero illumination, and γ is the system non-linearity. Even cameras with sensors that have an essentially linear response to light may produce images that have values adjusted with a value of γ other than one for the efficient use of the 0–255 range.

7.3. Direct use of captured images

Most inexpensive systems attempt to capture a relative estimate of Lambertian reflectance, expressed directly in terms of RGB. A Lambertian reflector reflects the same radiance in all directions for any incident energy flux density.

The Lambertian reflectance ρ_d is the fraction of incident energy reflected, and is related to the BRDF by:

$$f_r(\lambda, x, y, \boldsymbol{\omega}_i, \boldsymbol{\omega}_r) = \rho_d(\lambda, x, y)/\pi. \quad (4)$$

The radiance reflected for a Lambertian surface then is:

$$L_p(\lambda) = \rho_d(\lambda, x, y) \int L_s(\lambda, \boldsymbol{\omega}_s)\mathbf{n}_o \cdot \boldsymbol{\omega}_i \mathbf{n}_s \cdot \boldsymbol{\omega}_s \, \mathrm{d}A_s / r_s^2. \quad (5)$$

The reflected radiances measured at each pixel then are a good estimate of the relative spatial variation for Lambertian surfaces if $\mathbf{n}_o \cdot \boldsymbol{\omega}_i \mathbf{n}_s \cdot \boldsymbol{\omega}_s$ and r_s^2 are approximately the same for all points on the surface imaged at any given time. Maintaining constant r_s is relatively straightforward for systems with a fixed scanner location and object placed on a turntable. As long as the distance to the light source is large relative to the size of the surface area being imaged, the effect of varying r_s will be small. One approach to controlling the variation due to the changing incident angle is to use a large diffuse light source, so that each point on the surface is illuminated by nearly the entire hemisphere above it. Relying on indirect illumination in a room can achieve this effect. Alternatively, for systems that acquire texture simultaneously with range images, a camera flash can be used nearly collocated with the camera sensor (the standard design for a commodity camera). Surfaces obtained in each range image are oriented so that the surface normal is nearly parallel in the direction of the camera sensor. The captured points then will all be illuminated with a value of $\mathbf{n}_o \cdot \boldsymbol{\omega}_i \mathbf{n}_s \cdot \boldsymbol{\omega}_s$ close in value to 1.0. An additional advantage of using the flash built into the camera is that it is designed to be compatible with the spectral sensitivity of the camera sensor to produce good color match.

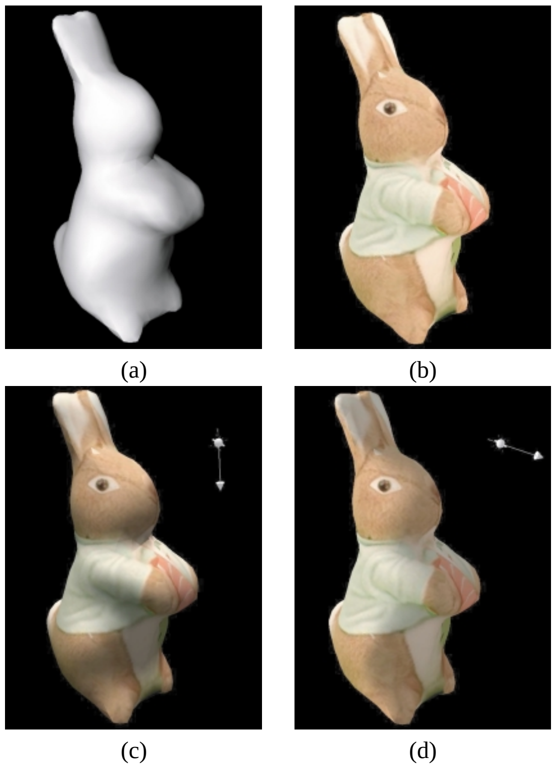

Figure 8: *An example of a texture-mapped model obtained from an inexpensive scanner; (a) the captured geometry; (b) texture displayed as-captured; (c) textured model relit from above; and (d) textured model relit from the back.*

Captured image data can represent rich appearance details as can be seen by contrasting the model shown in Figure 8(b) with a texture map with geometry alone Figure 8(a). The details of the fur can be seen in the texture, that would be essentially impossible to capture as geometry. However, there are clearly shadows on the bunny's coat that are fixed in the texture. Figures 8(c) and (d) show the model relit from novel directions. The texture looks flatter because the detail shadows do not appear consistent with the overall lighting direction.

7.4. Correcting captured images

While they produce approximations of the relative reflectance, inexpensive camera systems leave the texture pixels in the form given by (3). If data from such systems are to be used in rendering systems that use true physical parameters, a grayscale card should be used to estimate the γ of the color camera. A grayscale card image can also be used to assess the effect of the light source and camera spectral sensitivities on the RGB values. Absolute reflectance values can be estimated by capturing a reference white card with the object, or by obtaining separate spot measurements of the spectral reflectance of the object.

High-end systems that capture very accurate, dense range images, coupled with low noise high resolution color cameras may also be used to capture texture images. In these systems, images can be corrected using the geometric information to adjust for variations in angle and distance. Thresholding can be used to eliminate low values for values in shadow, and high values in specular highlights. Alternatively the geometry can be used to predict areas that will be in shadow or potentially in narrow specular peaks. Levoy *et al.* [95] describe the use of a CCD digital still camera with a laser stripe laser scanner to acquire accurate estimates of Lambertian reflectance.

7.5. Spatially uniform, directionally varying BRDF

An alternative to acquiring a spatially detailed map of BRDF that has no directional variation, is to acquire details of a directionally varying BRDF on objects with no spatial variation of surface properties. Such methods have been

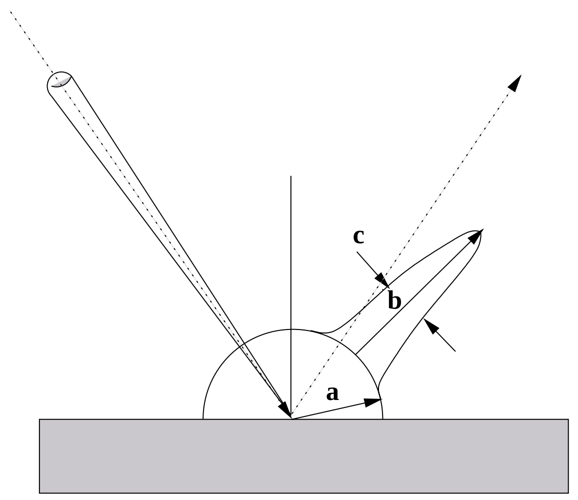

Figure 9: *Torrance–Sparrow inspired reflectance models attempt to model the magnitude of Lambertian reflected light* **a** *with a parameter ρ_d, the magnitude of directionally reflected light* **b** *with a parameter ρ_s and the width of the directional lobe* **c** *with a parameter σ.*

described by Ikeuchi and Sato [96] for a range and intensity image pair, and Baribeau *et al.* [87] for polychromatic laser data. These methods use systems in which the angle between sensor and light source position is fixed. However, because the scanner sees a uniform BRDF surface with a variety of surface orientations, data are obtained for $L_r(\lambda, \boldsymbol{\omega}_i, \boldsymbol{\omega}_r)$ for a variety of values of $(\boldsymbol{\omega}_i, \boldsymbol{\omega}_r)$. The methods compensate for not sampling the entire range of angles over the hemisphere by using the observed data to fit a parametric reflectance model. Each paper uses a version of the Torrance–Sparrow model [97]. Torrance–Sparrow-inspired models of BRDF are expressed generically as:

$$f_r(\lambda, \boldsymbol{\omega}_i, \boldsymbol{\omega}_r) = \rho_d(\lambda)/\pi + \rho_s(\lambda) g(\sigma, \boldsymbol{\omega}_i, \boldsymbol{\omega}_r) \quad (6)$$

where ρ_d is the fraction of incident light reflected diffusely (i.e. as a Lambertian reflector), ρ_s is the fraction of light reflected near the specular direction in excess of the diffusely reflected light in that direction, and g is a function that depends on a parameter σ characterizing surface roughness as well as the angles of incidence and reflection. Methods attempt to estimate the three parameters ρ_d, ρ_s and σ to give the shape of the reflectance function diagrammed in Figure 9.

For example, Ikeuchi and Sato [96] begin by assuming all pixels reflected diffusely, and estimate values of ρ_d and the light source direction (assumed uniform across the surface). This value is then refined by thresholding pixels which have values well above that predicted by the product of ρ_d and $\mathbf{n}_o \cdot \boldsymbol{\omega}_i$ (which result either from specular reflections or surface interreflections) and well below the predicted value (which result from either attached or cast shadows). After the estimates of ρ_d and $\boldsymbol{\omega}_i$ are made, an iterative process over non-Lambertian pixels distinguishes specular versus interreflection pixels based on observed angle relative to the angle of reflection. From the values of radiance recorded for specular pixel, values of the specular reflectance and surface roughness parameter are estimated. Alternatively Baribeau *et al.* [87] capture samples of BRDF for a variety of incident/reflected angle pairs using the polychromatic range sensor. These data are then fit to the parametric model using a non-linear least-squares algorithm.

These spatially uniform techniques of course do not require objects that are completely uniform, but objects with surfaces that can be segmented into reasonably large uniform areas.

7.6. Spatially and directionally varying BRDF

To capture both spatially and directionally varying BRDF, methods based on photometric stereo are used. Photometric stereo, introduced by Woodham [98] uses N images of an object from a single viewpoint under N different lighting conditions. Initially, photometric stereo was used to estimate surface normals, and from the normals surface shape. Assuming a Lambertian surface, and small light sources of uniform strength an equation for the surface normal $\mathbf{n}_o$ visible through each pixel p in each image m for each light source in direction $\boldsymbol{\omega}_{m,i}$ is given by:

$$\boldsymbol{\omega}_{m,i} \cdot \mathbf{n}_o = \xi G_{m,p} \quad (7)$$

where $G_{i,p}$ is the image grayscale value after correction for non-linear γ values, and ξ is a scaling constant that includes the light source radiance and subtended solid angle. Since $\mathbf{n}_o$ has unit length and thus represents only two independent variables, we can solve three equations for $\mathbf{n}_o$ and ξ.

Kay and Caelli [99] couple the idea of images from a photometric stereo system with a range image obtained from the same viewpoint to expand on the idea introduced by Ikeuchi and Sato. Rather than sampling a variety of directions by viewing many orientations across the surface, multiple incident light directions are observed for each surface point from the set of photometric images. Kay and Caelli used high dynamic range images to be able to capture specular objects by taking pairs of images for each lighting condition with and without a grayscale filter. Because the directional sampling is still sparse, the data are fit to a Torrance–Sparrow-inspired reflectance model. The fitting process proceeds in four passes. First, weights are estimated to account for noise in the surface and image data. Next, pixels are classified as to whether there is enough data to estimate the model parameters. In the third pass the parameters are estimated where data is adequate. In the final pass parameters are estimated for the areas in which there was insufficient data from the intensity maps. The only restriction on the technique is that interreflections are not accounted for, so strictly the method applies only to convex objects.

Sato *et al.* [100] presented a method for obtaining an estimate of BRDF for a full object. Range and color images are obtained for an object, with the object, sensor and light source positions registered by calibration by moving the object with a robot arm manipulator. After the full object is reconstructed, the color images—showing the object from a variety of views and illumination directions—are used to fit a Torrance–Sparrow-inspired model. The parameter fitting problem is simplified by separating diffusely and specularly reflected light in each image by examining the color of each point on the surface in various images. Assuming non-white, dielectric materials, the diffuse component will be the color of the object (i.e. the result of body reflection), while the specular component will be the color of the light source (i.e. the result of surface reflection) [101]. Because the specular component is sampled sparsely along the surface (there is no way to guarantee that a specular highlight will be obtained for each point even with a large number of images) the estimate of specular reflectance parameters are interpolated over larger areas of the object.

Lensch *et al.* [102] take a "top-down" approach to estimating spatially varying BRDF. High dynamic range images are taken from multiple views of an object of known shape. Initially, it is assumed that the pixel luminance all represent samples from a single BRDF. After this global BRDF is computed, two groups of luminances values are formed based on their distance from the global BRDF estimate, and two new BRDFs are computed. This splitting process is repeated until the distance of samples in a group from the BRDF computed from them falls to some threshold. Point by point variations are computed by computing each BRDF for each point on the final model as a linear combination of the set of BRDFs formed from the groups.

7.7. Capturing reflectance and small scale structure

Methods for obtaining texture may not just estimate reflectance, but may also capture small scale details at a resolution finer than the underlying range image. Rushmeier *et al.* [103] developed a photometric stereo system attached to a range imaging system. The photometric system allowed the calculation of normals maps on the surface at a higher spatial resolution than the underlying range image. They developed a method [104] to use the normals of the underlying low spatial resolution range image to adjust the images acquired by the photometric system to insure that the fine detail normals that are computed are consistent with the underlying mesh. Given the range images and detailed normals, the acquired color images were then adjusted to produce estimates of the Lambertian reflectance of the surface. Figure 10 shows an example of an underlying low resolution geometry sampled at approximately every 2 mm, and the same geometry with a normals map added to show detailed features every 0.5 mm.

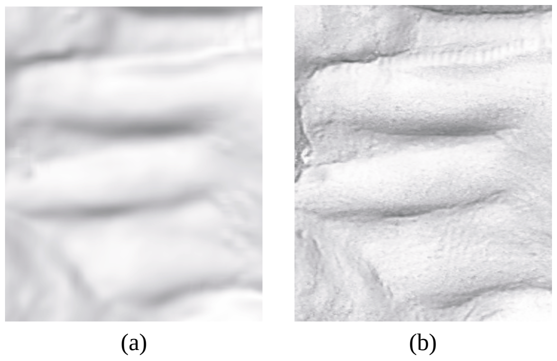

Figure 10: *An example of a normals map used to enhance the display of geometric detail. (a) Shows the underlying 2 mm resolution geometry. (b) Shows the geometry displayed with a 0.5 mm resolution normals map. The illumination is from a novel direction—i.e. not the direction of any of the illumination in any of the captured images.*

Dana *et al.* [105] observed that even full BRDF and normals maps are not adequate for capturing the change in detail surface appearance with lighting and view for surfaces with fine scale geometric complexity such as bread and velvet. They developed the concept of bidirectional textures, which are sets of images (rather than individual values) of surfaces for varying light and viewpoint.

No scanning method has been developed to truly capture bidirectional textures for complete objects. However, there have been a number of techniques that use the concept of view dependent texture maps. View dependent texture maps were introduced by Debevec *et al.* [106] in the context of building models from photogrammetry and generic parameterized models. A different texture map, obtained from points closest to the current view, is used for each view of a model. View dependent texture maps can portray the variation of surface appearance due to changes in self-occlusion as well as BRDF. View dependent texture maps as described in [106] are not varied for different lighting conditions. Pulli *et al.* [107] applied the idea to texturing range images. In an interactive viewer, only the range and color images that would be visible from the current view are used. Texture is synthesized on the fly using a combination of the three acquired textures closest to the current view. The effect is to render the effects of BRDF, occlusion and shadowing for the lighting conditions that existing during acquisition. Since the textures were acquired with both lighting and view changing, the effect is approximately the same as observing the object with a headlight at the viewer position.

Miller *et al.* [108] developed the idea of surface light fields that represent the light leaving each point on a surface in all directions. They applied this idea to synthetic models. Nishino *et al.* [109] developed the Eigen-Texture system to capture and represent surface light field data. In their method, a light is fixed to the object coordinate system, and M views of an object are obtained using a turntable. The result is M small texture maps for each triangle on a simplified version of the geometry obtained from the range images. The series of M small texture maps are compressed by performing an eigenstructure analysis on the series and finding a small number of textures that can be used as an approximate basis set to form textures in the view space encompassed by the originally M textures. The textures then represent the effects of BRDF, self-shadowing, and self-occlusion effects for the single lighting condition. Eigen-Textures obtained for many different lighting conditions can be combined linearly to generate textures for novel lighting conditions. Wood *et al.* [110] proposed an alternate method for capturing and storing surface light fields using a different approach for data compression. They also demonstrated how small changes could be made in an object represented by surface light fields while maintaining a plausible, if not completely accurate, appearance.

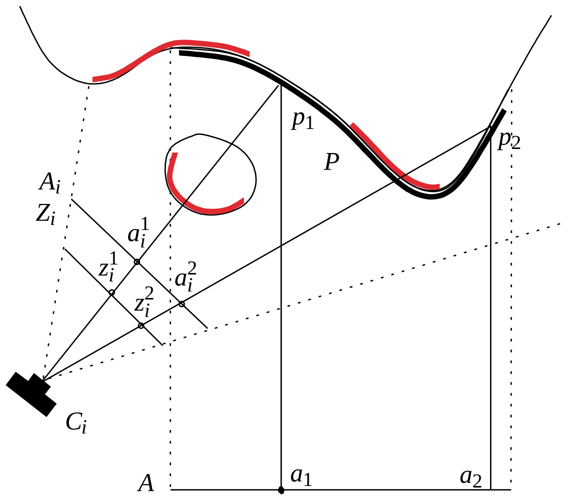

Figure 11: *In determining which parts of a captured texture image A_i can be used in a texture map A for a surface P, occlusion effects must be accounted for. Here the captured texture pixel a_i^1 should not appear in the final texture map pixel a_i because the point p_i is occluded from the point of view of camera C_i.*

8. Texture Map Reconstruction

Texture map reconstruction involves combining all the texture maps acquired for an object into a single non-redundant map over the entire object. Texture map reconstruction may start with meshes that store a color for each vertex point, and form images. Other methods begin with acquired (and possibly processed) images. Methods for texture map reconstruction starting with images may either select one piece from one acquired image to texture each surface area or they may combine multiple maps that cover each surface area.

Soucy *et al.* [76] developed a method for generating a texture map from color per vertex models. The dense triangle mesh is simplified to reduce the triangle count. Barycentric coordinates are saved for each color triplet for which the vertex has been removed. Separate texture maps are created for each triangle in the simplified mesh. The texture image for each triangle is required to be a half-square triangle. Appropriate colors are assigned to texture pixels using the original vertex colors and their barycentric coordinates. Continuity between the texture maps is insured by requiring vertices to coincide with pixel centers in the texture map, and by requiring the number of pixels along the edges of maps for adjacent texture maps to be integer multiples of one another. With this constraint, pixels representing the same location on two different texture maps can be forced to have identical values. All of the individual texture maps are then packed into a single texture image.

Methods for reconstructing texture from sets of images have in common that for each texture image, the triangles visible in that image are identified. As shown in Figure 11, simply checking that a surface is contained within the image view frustum and is oriented toward the camera position is not adequate. A full rendering of the model is required to detect whether another surface occludes the surface being mapped.

Methods for reconstructing non-redundant texture for inexpensive scanner systems that use captured images directly for building maps generally select a piece of a single image for each triangle in the mesh. An example of this sort of method is described by Matsumoto *et al.* [111]. There are two desirable properties in selecting the image that contributes the texture for a given triangle—it should be from the viewpoint in which the triangle projects to the largest area, and it should be from the same image as adjacent triangles. Matsumoto *et al.* cast this as an energy minimization problem, where the energy is defined as the difference between a penalty function expressing the distance between the images used for adjacent triangles and the scaled projected area of a triangle on an an image.

An example of a texture map produced by an inexpensive scanning system that selects image segments as large as possible and then packs them into a single texture map is shown in Figure 12 for the model that was shown in Figure 8.

Individual textures may be selected for regions of the surface encompassing multiple triangles. Rocchini *et al.* [2] describe a method for selecting one source texture per region, with regions covered by a single source map made

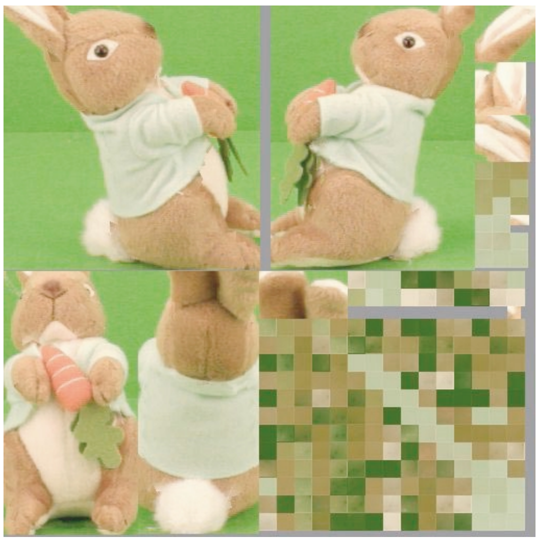

Figure 12: *An example of the texture image used to display the model in Figure* 8*.*

as large as possible. First, a list of images containing each vertex is found. Then in an iterative procedure, regions are grown so that large regions of the surface are mapped to the same image. The problem remains then of adjusting the boundaries between regions so seams are not visible. For the triangles on boundaries between different source images, a detailed local registration is performed so that details from the two source texture images match.

Methods that use zippering for mesh integration use the original texture map for each section of range image used in the final mesh. Just as overlapping meshes are used to adjust point positions to reduce line-of-sight errors, overlapping textures are used to adjust texture values to eliminate abrupt color changes in the texture. Texture in the overlap region is the weighted average of the two overlapping textures, with the weight of each texture decreasing with distance to the edge of the corresponding range image. Figures 13(a) and (b) show two overlapping scans to be merged. Figure 13(c) shows the result after the geometries have been zippered (or stitched) together, with the original texture maps. Figure 13(d) shows the final result after the texture in the overlap region has been adjusted.

Rather than just use multiple textures pair wise, other methods use data from multiple textures that contain each triangle. Such methods are successful and avoid ghosting and blurring artifacts if they are preceded by registration techniques that make use of texture image data. Johnson and Kang [32] use all textures containing each triangle, with a weighted average that uses the angle of the surface normal to the direction to camera for each image as the weight. In Pulli *et al.* 's [107] view-dependent texturing uses

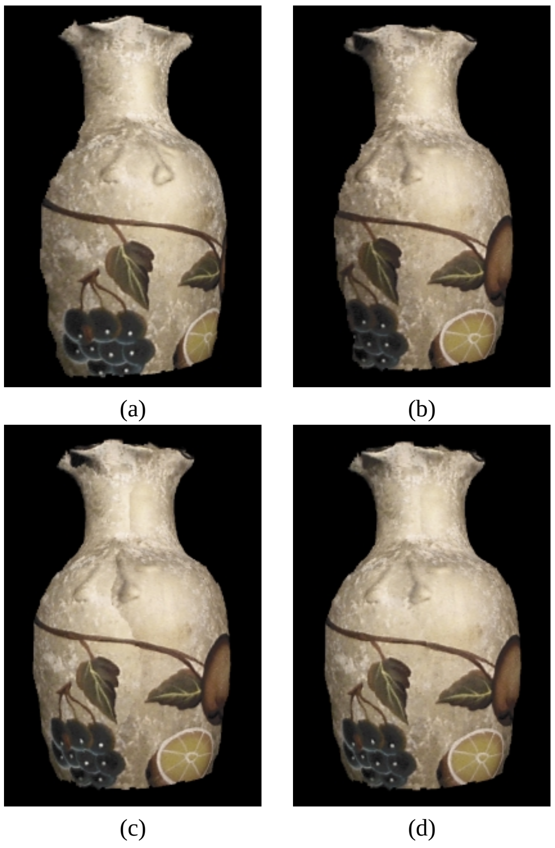

Figure 13: *An example of the zippering approach to combining texture maps: (a) and (b) show two input scans to be merged; (c) shows the merged textures without adjustment; (d) shows the final texture after adjustment.*

three types of weights in combining three source textures. First a weight representing the angle between the current view and each source view is computed. Then, similar to Johnson and Kang, the surface normal to view angle is used. Finally, similar to the zippering methods, these weights are combined with a weight that decreases with distance to the texture edge. Neugebauer and Klein [90] combine multiple textures using weights that account for the angle of the surface normal and view direction, and the distance to the edge of the region of a texture image that will be used in the final model. Because they use images that may still contain artifacts such as specular highlights, Neugebauer and Klein use a third weight that eliminates outliers.

Bernardini *et al.* [38] describe a method that uses all available maps representing reflectance and normals at each triangle obtained using the system described in [103]. To minimize color variations, before the maps representing reflectance are combined, a global color balance is performed [104]. A set of points are randomly sampled on the integrated surface mesh. All of the maps representing reflectance are projected onto the integrated mesh, and for each point all of the maps that contain the point are identified. A set of linear equations is formed for a scalar correction factor for each channel for each image that sets the colors in each map representing a common point equal. A least squares solution is performed to compute the correction factors for this over determined system. The normals maps were previously made consistent with one another by the process used to make them consistent with the underlying integrated mesh. The map values are then combined using three weights. Similar to the other methods, one is based on the area of the triangle in the image using the dot product of normal to view direction, combined with the distance from the camera sensor to the triangle. Another is a weight which diminishes with distance to the edge of the texture. Finally a third weight is used that indicates whether it was possible to compute a normal from the photometric images, or if the normal from the underlying integrated mesh was used.

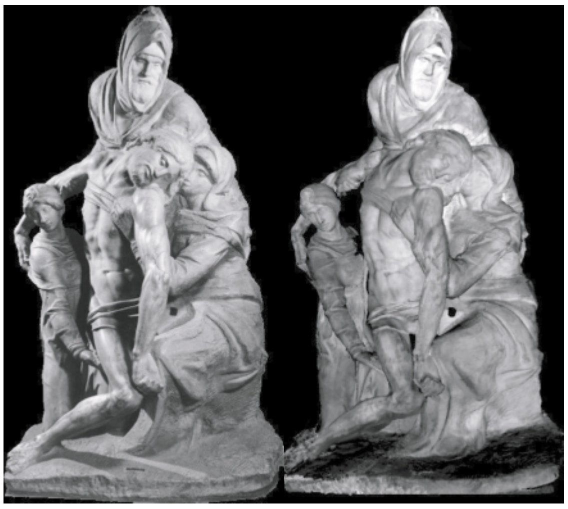

Figure 14: *(left) A photograph of Michelangelo's Florentine Pietà. (right) A synthetic picture from the 3D computer model.*

9. Scanning Systems and Projects

By combining different features from the various methods for each step outlined in Figure 1, it is possible to compose many different systems for producing a 3D model of an existing object suitable for computer graphics modeling and rendering. The design of the particular processing pipeline depends on the requirements of the end application, constrained by the budgetary limitations for acquiring the data.

A number of scanning applications with emphasis on graphic display as the end product have been documented. Major application areas include scanning historical objects for scholarly study and virtual museums, scanning of humans and e-commerce.

The National Research Council of Canada has conducted a series of projects over the past 15 years scanning historical artifacts ranging from 3D representations of oil paintings to archeological sites. Their experiences acquiring and displaying geometry and color reflectance of a variety of objects are described in various publications [112]. In particular Beraldin *et al.* [113] present a detailed practical discussion of using a portable scanner (i.e. suitcase-sized) to scan a number of sculptural and architectural features on site in Italy. As an example of current capabilities, they describe the scanning of Pisano's Madonna col Bambino in the Cappella degli Scrovegni in Padova. The were able to acquire 150 scans at 1 mm resolution of the approximately 1 m tall statue in a 7 h period. The range images were registered and integrated using Polyworks^{TM} software.

Many other cultural heritage projects are ongoing or recently completed. Zheng, of the Kyushu Institute of Technology in collaboration with the Museum of Qin Shihuang Terra Cotta Warriors and Horses is conducting an extensive scanning project to build models of relics found at the site [114]. A custom portable laser scanner coupled with a digital video camera was designed for the project. Besides presenting the models as they are, the project seeks to facilitate piecing together damaged relics, and digitally restoring full color to figures using pigment fragments that have been found.

Ikeuchi *et al.* have developed many techniques for the steps in the model acquisition pipeline. These techniques are now being applied to building a model of the 13 m tall Kamakura Buddha from color images and time-of-flight range scanning data [115].

Levoy *et al.* recently used a combination of laser triangulation range scanning and high-resolution digital color imaging to acquire models of many of the major works of Michelangelo [95]. The high-end equipment employed produced large quantities of data. To make the results usable, they developed a novel rendering system that generates images directly from points rather than from triangle primitives [116].

Bernardini *et al.* [117] used a lower resolution structured light system coupled with a photometric lighting system for higher resolution reflectance and normals maps to scan Michelangelo's Florentine Pietà. A rendering of the model is shown next to a photograph of the statue in Figure 14.

Several projects are addressing the scanning of human shape, e.g. [118]. Many of these applications address purely geometric issues such as fit and ergonomic design, rather than preparing models for computer graphics display. For animation systems however, there has been a great deal of interest in the scanning of human faces. Building a realistic face model is one of the most demanding applications, because of human familiarity with the smallest details of the face. Yau [119] described a system for building a face model from a range scan that used light striping and a color image for texture mapping. Nahas *et al.* [120] describe obtaining a realistic face model from a laser range scanner that captures reflectance information as well. Marschner *et al.* [121] described a method to obtain skin BRDF for realistic faces using color images and a detailed model from a range scanner, in a method similar to that used by Ikeuchi and Sato [96]. This work was extended to spatially varying skin reflectance [122].

Debevec *et al.* [123] designed a specialized rig for obtaining hundreds of images of an individual face with calibrated lighting. They use this data to compute spatially varying BRDFs and normals that are mapped onto a lower resolution model of the face that is obtained with a structured light system. Haro *et al.* [124] describe a less rigorous, but also much less expensive, method for obtaining detailed facial geometry. Photometric stereo is used to capture the geometry of small patches of skin impressions made in a polymeric material. These patches are placed onto appropriate areas of the face model, and grown using texture synthesis techniques to cover the whole face.

The cultural heritage and human face applications discussed above have emphasized using relatively high-end systems. An emerging application for acquired 3D models is e-commerce—using 3D models to allow shoppers to examine and/or customize items for purchase over the internet. This new application requires both inexpensive equipment, and a much higher level of "ease-of-use." Companies targeting this application area are offering systems at relatively low (<$10,000) price for scanning small objects.

10. Conclusions

The current state of the art allows the acquisition of a large class of objects, but requires expert operators and time consuming procedures for all but the simplest cases. Research is needed to improve the acquisition pipeline in several key aspects:

- planning methods for data acquisition;
- reliable capture and robust processing of data for a larger class of objects, including large size objects, environments, and objects with challenging surface properties;
- automation of all the steps, to minimize user input;
- real-time feedback of the acquired surface;
- improved capture and representation of surface appearance;
- methods for assessing global model accuracy after range scan registration.

Scanning and reconstruction technology will enable a more extensive use of 3D computer graphics in a wide range of applications.

References

1. T. Várady, R. R. Martin and J. Cox. Reverse engineering of geometric models—an introduction. *Computer Aided Design*, 29(4):255–268, 1997.
2. C. Rocchini, P. Cignoni, C. Montani and R. Scopigno. Multiple textures stitching and blending on 3D objects. In *Proceedings of the 10th Eurographics Workshop on Rendering*, Granada, Spain: pp. 127–138. June, 1999.
3. M. Polleyfeys, R. Koch, M. Vergauwen and L. V. Gool. Hand-held acquisition of 3D models with a video camera. In *Proceeding of the 2nd International Conference on 3D Digital Imaging and Modeling*, Ottawa, Canada: pp. 14–23. October, 1999.
4. J. Y. Zheng. Acquiring 3D models from sequences of contours. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(2):163–178, 1994.
5. J.-A. Beraldin, S. F. El-Hakim and F. Blais. Performance evaluation of three active vision systems built at the National Research Council of Canada. In *Proceedings of the Optical 3D Measurement Techniques III*, NRC Technical Report 39165. Vienna: pp. 352–361. 1995.
6. C. Zitnick and J. A. Webb. Multi-baseline stereo using surface extraction. In *Technical Report CMU-CS-96-196*. Computer Science Department, Carnegie Mellon University, Pittsburgh, PA, 1996.
7. P. Boulanger. Knowledge representation and analysis of range data, *Tutorial Notes, Proceedings of the 2nd International Conference on 3D Digital Imaging and Modeling*, 1999.
8. G. M. Cortelazzo, C. Doretto and L. Lucchese. Free-form textured surfaces registration by a frequency domain technique. In *International Conference on Image Processing, ICIP '98*, pp. 813–817. 1998.
9. G. Roth. Registering two overlapping range images. In *Proceeding of the 2nd International Conference on 3D Digital Imaging and Modeling*, Ottawa, Canada: pp. 191–200. October, 1999.
10. D. Zhang and M. Hebert. Harmonic maps and their applications in surface matching. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR '99)*, pp. 524–530. 1999.

11. A. E. Johnson and M. Hebert. Using spin images for efficient object recognition in cluttered 3D scenes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(5):433–449, May, 1999.

12. M. A. Greenspan and P. Boulanger. Efficient and reliable template set matching for 3D object recognition. In *Proceedings of the 2nd International Conference on 3D Digital Imaging and Modeling (3DIM)*, pp. 230–239. 1999.

13. P. J. Besl and N. D. McKay. A method for registration of 3D shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(2):239–256, February, 1992.

14. Y. Chen and G. G. Medioni. Object modeling by registration of multiple range images. *Image and Vision Computing*, 10(3):145–155, 1992.

15. Z. Zhang. Iterative point matching for registration of free-form curves and surfaces. *International Journal of Computer Vision*, 13(2):119–152, 1994.

16. C. Dorai, J. Weng and A. K. Jain. Optimal registration of object views using range data. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(10):1131–1138, October, 1997.

17. C. Dorai, G. Wang, A. K. Jain and C. Mercer. Registration and integration of multiple object views for 3D model construction. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(1):83–89, January, 1998.

18. D. W. Eggert, A. W. Fitzgibbon and R. B. Fisher. Simultaneous registration of multiple range views for use in reverse engineering of CAD models. *Computer Vision and Image Understanding*, 69(3):253–272, March, 1998.

19. B. K. P. Horn. Closed form solutions of absolute orientation using unit quaternions. *Journal of the Optical Society of America*, 4(4):629–642, April, 1987.

20. R. M. Haralick, H. Joo, C. Lee, X. Zhuang, V. G. Vaidya and M. B. Kim. Pose estimation from corresponding point data. *IEEE Transactions on Systems, Man and Cybernetics*, 19(6):1426–1446, 1989.

21. T. Masuda, K. Sakaue and N. Yokoya. Registration and integration of multiple range images for 3D model construction. In *Proceeding of ICPR '96*, IEEE. pp. 879–883. 1996.

22. G. Turk and M. Levoy. Zippered polygon meshes from range images. In A. Glassner (ed), *Proceedings of SIGGRAPH 94*, Computer Graphics Proceedings, Annual Conference Series. pp. 311–318. July, 1994.

23. R. Bergevin, M. Soucy, H. Gagnon and D. Laurendeau. Towards a general multiview registration technique. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(5):540–547, May, 1996.

24. R. Benjemaa and F. Schmitt. Fast global registration of 3D sampled surfaces using a multi-z-buffer technique. In *Proceedings of the International Conference on Recent Advances in 3D Digital Imaging and Modeling*, Ottawa, Canada: pp. 113–120. May, 1997.

25. K. Pulli. Multiview registration for large data sets. In *Proceedings of the 2nd International Conference on 3D Digital Imaging and Modeling*, Ottawa, Canada: pp. 160–168. October, 1999.

26. G. Blais and M. D. Levine. Registering multiview range data to create 3D computer objects. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(8):820–824, August, 1995.

27. P. J. Neugebauer. Reconstruction of real-world objects via simultaneous registration and robust combination of multiple range images. *International Journal of Shape Modeling*, 3(1 & 2):71–90, 1997.

28. A. J. Stoddart and A. Hilton. Registration of multiple point sets. In *Proceedings of the 13th International Conference on Pattern Recognition*, Vienna, Austria: pp. B40–B44. 1996.

29. E. Gagnon, J.-F. Rivest, M. Greenspan and N. Burtnyk. A computer-assisted range image registration system for nuclear waste cleanup. *IEEE Transactions on Instrumentation and Measurement*, 48(3):758–762, 1999.

30. R. C. Gonzalez and R. E. Woods. *Digital Image Processing*. Addison-Wesley, Reading, MA, 1993.

31. F. Bernardini and H. Rushmeier. Strategies for registering range images from unknown camera positions. In *Three-Dimensional Image Capture and Applications III*, (*Proceedings of SPIE* 3958). pp. 200–206. 2000.

32. A. Johnson and S. Kang. Registration and integration of textured 3D data. In *Proceedings of the International Conference on Recent Advances in 3D Digital Imaging and Modeling*, Ottawa, Canada: pp. 234–241. May, 1997.

33. A. Johnson and S. Kang. Registration and integration of textured 3D data. In *Technical Report CRL 96/4*. DEC-CRL. September, 1996.

34. C. Schütz, T. Jost and H. Hügli. Multi-feature matching algorithm for free-form 3D surface registration. In *Proceedings of the International Conference on Pattern recognition*. Brisbane, Australia: August, 1998.

35. S. Weik. Registration of 3D partial surface models using luminance and depth information. In *Proceedings of the International Conference on Recent Advances in 3D Digital Imaging and Modeling*, Ottawa, Canada: pp. 93–100. May, 1997.

36. K. Pulli. Surface reconstruction and display from range and color data, *PhD Thesis*, Department of Computer Science and Engineering, University of Washington, 1997.

37. R. Szeliski and H. Shum. Creating full panoramic mosaics and environment maps. In *Proceedings of SIGGRAPH 97,* Computer Graphics Proceedings, Annual Conference Series. pp. 251–258. 1997.

38. F. Bernardini, I. Martin and H. Rushmeier. High-quality texture reconstruction. *IEEE Transactions on Visnalization and Computer Graphics*, 7(4):318–332, 2001.

39. M. Rutishauser, M. Stricker and M. Trobina. Merging range images of arbitrarily shaped objects. In *Proceedings of CVPR '94*, pp. 573–580. 1994.

40. P. Hébert, D. Laurendeau and D. Poussart. Scene reconstruction and description: geometric primitive extraction from multiple view scattered data. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 286–292. 1993.

41. M. Soucy and D. Laurendeau. A general surface approach to the integration of a set of range views. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(4):344–358, April, 1995.

42. R. M. Bolle and B. C. Vemuri. On surface reconstruction methods. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(1):1–13, 1991.

43. R. Mencl and H. Müller. Interpolation and approximation of surfaces from three-dimensional scattered data points. In *State of the Art Report (STAR), Eurographics '98*. 1998.

44. F. Bernardini, C. Bajaj, J. Chen and D. Schikore. Automatic reconstruction of 3D CAD models from digital scans. *International Journal of Computational Geometry and Applications*, 9(4 & 5):327–370, August–October, 1999.

45. H. Edelsbrunner. Shape reconstruction with delaunay complex. In A. V. Moura and C. L. Lucchesi (eds), *LATIN'98: Theoretical Informatics. Third Latin American Symposium, Campinas, Brazil*, Lecture Notes in Computer Science, LNCS 1380. New York: Springer, pp. 119–132. 1998.

46. H. Edelsbrunner and E. P. Mücke. Three-dimensional alpha shapes. *ACM Transactions on Graphics*, 13(1):43–72, January, 1994.

47. C. Bajaj, F. Bernardini and G. Xu. Automatic reconstruction of surfaces and scalar fields from 3D scans. In *Proceedings of SIGGRAPH 95,* Computer Graphics Proceedings, Annual Conference Series. pp. 109–118. 1995.

48. H. Edelsbrunner. Surface reconstruction by wrapping finite sets in space. In *Technical Report 96-001*. Raindrop Geomagic Inc., 1996.

49. N. Amenta, M. Bern and M. Kamvysselis. A new Voronoi-based surface reconstruction algorithm. In *Proceedings of SIGGRAPH 96,* Computer Graphics Proceedings, Annual Conference Series. pp. 415–412. July, 1998.

50. N. Amenta and M. Bern. Surface reconstruction by Voronoi filtering. *Discrete Computational Geometry*, 22(4):481–504, 1999.

51. N. Amenta, S. Choi and R. Kolluri. The power crust. In *Proceedings of the 6th ACM Symposium on Solid Modeling and Applications*. 2001.

52. N. Amenta, S. Choi and R. Kolluri. The power crust, unions of balls, and the medial axis transform. *International Journal of Computational Geometry and its Applications* (special issue on surface reconstruction, in press).

53. N. Amenta, S. Choi, T. K. Dey and N. Leekha. A simple algorithm for surface reconstruction. In *Proceedings of 16th ACM Symposium on Computational Geometry*, pp. 213–222. 2000.

54. T. K. Dey, J. Giesen and J. Hudson. Delaunay based shape reconstruction from large data. In *Proceedings of IEEE Visualization 2001*. 2001, (in press).

55. F. Bernardini and C. Bajaj. Sampling and reconstructing manifolds using alpha-shapes. In *Proceedings of the 9th Canadian Conference on Computational Geometry*, pp. 193–198. August, 1997. Updated online version available at `www.qucis.queensu.ca/cccg97`.

56. D. Attali. r-regular shape reconstruction from unorganized points. *Computational Geometry: Theory and Applications*, 10:239–247, 1998.

57. T. K. Dey, K. Mehlhorn and E. A. Ramos. Curve reconstruction: connecting dots with good reason. *Computational Geometry: Theory and Applications*, 15:229–244, 2000.

58. F. Bernardini, J. Mittleman, H. Rushmeier, C. Silva and G. Taubin. The ball-pivoting algorithm for surface reconstruction. *IEEE Transactions on Visualization and Computer Graphics*, 5(4):349–359, October–December, 1999.

59. M. Gopi, S. Krishnan and C. T. Silva. Surface reconstruction based on lower dimensional localized Delaunay triangulation. In *Proceedings of Eurographics 2000*, pp. 467–478. 2000.

60. B. Curless and M. Levoy. A volumetric method for building complex models from range images. In *Proceedings of SIGGRAPH 96*, *ACM SIGGRAPH*, Computer Graphics Proceedings, Annual Conference Series. pp. 303–312. August, 1996.

61. M. Wheeler, Y. Sato and K. Ikeuchi. Consensus surfaces for modeling 3D objects from multiple range images. In *Sixth International Conference on Computer Vision*, IEEE. pp. 917–924. 1998.

62. A. Hilton, A. Stoddart, J. Illingworth and T. Windeatt. Reliable surface reconstruction from multiple range images. In *Fourth European Conference on Computer Vision*, pp. 117–126. 1996.

63. W. Lorensen and H. Cline. Marching cubes: a high resolution 3D surface construction algorithm. *Computer Graphics*, 21(4):163–170, 1987.

64. J.-D. Boissonnat and F. Cazals. Smooth surface reconstruction via natural neighbour interpolation of distance function. In *Proceedings of the 16th ACM Symposium on Computational Geometry*, pp. 223–232. 2000.

65. K. Pulli, T. Duchamp, H. Hoppe, J. McDonald, L. Shapiro and W. Stuetzle. Robust meshes from multiple range maps. In *Proceedings of the International Conference on Recent Advances in 3D Digital Imaging and Modeling*, Ottawa, Canada: pp. 205–211. May, 1997.

66. H. Hoppe, T. DeRose, T. Duchamp, J. McDonald and W. Stuetzle. Mesh optimization. In *Proceedings of SIGGRAPH '93*, Computer Graphics Proceedings, Annual Conference Series. pp. 19–26. 1993.

67. M.-E. Algorri and F. Schmitt. Surface reconstruction from unstructured 3D data. *Computer Graphics Forum*, 15(1):47–60, 1996.

68. J. Neugebauer and K. Klein. Adaptive triangulation of objects reconstructed from multiple range images. In *IEEE Visualization '97, Late Breaking Hot Topics*. 1997.

69. M. Reed and P. Allen. 3D modeling from range imagery: an incremental method with a planning component. *Image and Vision Computing*, (17):99–111, 1999.

70. C. Rocchini, P. Cignoni, F. Ganovelli, C. Montani, P. Pingi and R. Scopigno. Marching intersections: an efficient resampling algorithm for surface management. In *Proceedings of the International Conference on Shape Modeling and Applications (SMI 2001)*. Genova, Italy: 7–11 May, 2001.

71. D. Terzopoulos, A. Witkin and M. Kass. Constraints on deformable models: recovering 3D shape and nonrigid motion. *Artificial Intelligence*, 36:91–123, 1988.

72. A. Pentland and S. E. Sclaroff. Closed form solutions for physically based shape modelling and recovery. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(7):715–729, 1991.

73. R. T. Whitaker. A level-set approach to 3D reconstruction from range data. *International Journal of Computer Vision*, 29(3):203–231, 1998.

74. J. Gomes and O. Faugeras. Level sets and distance functions. In *Proceedings of the 6th European Conference on Computer Vision*, pp. 588–602. 2000.

75. M. Garland. Multiresolution modelling: survey and future opportunities. In *State of the Art Report (STAR), Eurographics '99*. 1999.

76. M. Soucy, G. Godin and M. Rioux. A texture-mapping approach for the compression of colored 3D triangulations. *The Visual Computer*, 12:503–513, 1996.

77. J. Maillot, H. Yahia and A. Verroust. Interactive texture mapping. In *Proceedings of SIGGRAPH 93*, Computer Graphics Proceedings, Annual Conference Series. pp. 27–34. 1993.

78. M. Eck, T. DeRose, T. Duchamp, H. Hoppe, M. Lounsbery and W. Stuetzle. Multiresolution analysis of arbitrary meshes. In R. Cook (ed), *Proceedings of SIGGRAPH 95*, Computer Graphics Proceedings, Annual Conference Series. pp. 173–182. August, 1995.

79. S. R. Marschner. Inverse rendering for computer graphics, *PhD Thesis*, Cornell University, 1998.

80. P. Sloan, D. Weinstein and J. Brederson. Importance driven texture coordinate optimization. *Computer Graphics Forum*, 17(3):97–104, 1998. Proceedings of EUROGRAPHICS '98.

81. H. Hoppe, T. DeRose, T. Duchamp, M. Halstead, H. Jin, J. McDonald, J. Schwitzer and W. Stuelzle. Piecewise smooth surface reconstruction. In *Proceedings of SIGGRAPH 94*, Computer Graphics Proceedings, Annual Conference Series. pp. 295–302. 1994.

82. C. Loop. Smooth subdivision surfaces based on triangles, *MS Thesis*, Department of Mathematics. University of Utah, August, 1987.

83. M. Eck and H. Hoppe. Automatic reconstruction of B-splines surfaces of arbitrary topological type. In *Proceedings of SIGGRAPH 96*, Computer Graphics Proceedings, Annual Conference Series. pp. 325–334. 1996.

84. H. Hoppe, T. DeRose, T. Duchamp, J. McDonald and W. Stuelzle. Surface reconstruction from unorganized points. *Computer Graphics*, 26(2):71–78, July, 1992. Proceedings of SIGGRAPH 92.

85. J. Peters. Constructing C^1 surfaces of arbitrary topology using biquadratic and bicubic splines. In N. Sapidis (ed), *Designing Fair Curves and Surfaces*, SIAM, pp. 277–293. 1994.

86. V. Krishnamurthy and M. Levoy. Fitting smooth surfaces to dense polygon meshes. In *Proceedings of SIGGRAPH 96, ACM SIGGRAPH*, Computer Graphics Proceedings, Annual Conference Series. pp. 313–324. August, 1996.

87. R. Baribeau, M. Rioux and G. Godin. Color reflectance modeling using a polychromatic laser range sensor. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 263–269, 1992.

88. K. Price. Computer vision bibiliography, `http://iris.usc.edu/Vision-Notes/bibliography/contents.html`.

89. R. Y. Tsai. An efficient and accurate camera calibration technique for 3D machine vision. *Computer Vision and Pattern Recognition*, 364–374, June, 1986.

90. P. Neugebauer and K. Klein. Texturing 3D models of real world objects from multiple unregistered photographics views. *Computer Graphics Forum*, 18(3):C245–C256, 1999. Proceedings of EUROGRAPHICS ’99.

91. H. P. A. Lensch, W. Heidrich and H.-P. Seidel. Automated texture registration and stitching for real world models. In W. Wang, B. A. Barsky and Y. Shinagawa (eds), *Proceedings of the 8th Pacific Conference on Computer Graphics and Applications*, pp. 317–326. 2000. IEEE Computer Society.

92. K. Nishino, Y. Sato and K. Ikeuchi. Appearance compression and synthesis based on 3D model for mixed reality. In *Proceeding of ICCV ’99*, Vol. 1. pp. 38–45. September, 1999.

93. P. Viola. Alignment by maximization of mutual information, *PhD Thesis*, MIT AI-Lab, June, 1995.

94. K. Matsushita and T. Kaneko. Efficient and handy texture mapping on 3D surfaces. *Computer Graphics Forum*, 18(3):C349–C357, 1999. Proceedings of EUROGRAPHICS ’99.

95. M. Levoy *et al*. The digital Michelangelo project: 3D scanning of large statues. In *Proceedings of SIGGRAPH 00*, Computer Graphics Proceedings, Annual Conference Series. pp. 131–144. 2000.

96. K. Ikeuchi and K. Sato. Deterimining reflectance properties of an object using range and brightness images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(11):1139–1153, 1991.

97. K. Torrance and E. Sparrow. Theory for off-specular reflection for roughened surface. *Journal of the Optical Society of America*, 57:1105–1114, September, 1967.

98. R. J. Woodham. Photometric method for determining surface orientation from multiple images. *Optical Engineering*, 19:139–144, 1980.

99. G. Kay and T. Caelli. Inverting an illumination model from range and intensity maps. *CVGIP—Image Understanding*, 59(2):183–201, 1994.

100. Y. Sato, M. Wheeler and K. Ikeuchi. Object shape and reflectance modeling from observation. In *Proceedings of SIGGRAPH 97*, Computer Graphics Proceedings, Annual Conference Series. pp. 379–388. 1997.

101. G. Klinker, S. Shafer and T. Kanade. Using a color reflection model to separate highlights from object color. In *International Conference on Computer Vision*, pp. 145–150. 1987.

102. H. P. A. Lensch, J. Kautz, M. Goesele, W. Heidrich and H.-P. Seidel. Image-based reconstruction of spatially varying materials. In K. Myzkowski and S. Gortler (eds), *Rendering Techniques ’01 (Proceedings of the 12th Eurographics Rendering Workshop)*. 2001.

103. H. Rushmeier, F. Bernardini, J. Mittleman and G. Taubin. Acquiring input for rendering at appropriate level of detail: digitizing a Pietà. In *Proceedings of the 9th Eurographics Workshop on Rendering*, Vienna, Austria: pp. 81–92. June, 1998.

104. H. Rushmeier and F. Bernardini. Computing consistent normals and colors from photometric data. In *Proceeding of the 2nd International Conference on 3D Digital Imaging and Modeling*, Ottawa, Canada: pp. 99–108. October, 1999.

105. K. Dana, B. van Ginneken, S. Nayar and J. Koenderink. Reflectance and texture of real-world surfaces. *ACM Transactions on Graphics*, 1:34, 1999.

106. P. Debevec, C. Taylor and J. Malik. Modeling and rendering architecture from photographs: a hybrid geometry- and image-based approach. In *Proceedings of SIGGRAPH 96*, Computer Graphics Proceedings, Annual Conference Series. pp. 11–20. August, 1996.

107. K. Pulli, M. Cohen, T. Duchamp, H. Hoppe, L. Shapiro and W. Stuetzle. View-based rendering: visualizing real objects from scanned range and color data. In *Proceedings of the 8th Eurographics Workshop on Rendering*, St Etienne, France: pp. 23–34. June, 1997.

108. G. S. P. Miller, S. Rubin and D. Ponceleon. Lazy decompression of surface light fields for precomputed global illumuniation. In *Proceedings of the 9th Eurographics Workshop on Rendering*, Vienna, Austria: pp. 281–292. June, 1998.

109. K. Nishino, Y. Sato and K. Ikeuchi. Eigen-texture method: appearance compression based on 3D model. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 618–624. June, 1999.

110. D. N. Wood, D. I. Azuma, K. Aldinger, B. Curless, T. Duchamp, D. H. Salesin and W. Stuetzle. Surface light fields for 3D photography. In *Proceedings of SIGGRAPH 00*, Computer Graphics Proceedings, Annual Conference Series. pp. 287–296. 2000.

111. Y. Matsumoto, H. Terasaki, K. Sugimoto and T. Arakawa. A portable three-dimensional digitizer. In *Proceedings of the International Conference on Recent Advances in 3D Digital Imaging and Modeling*, Ottawa, Canada: pp. 197–204. May, 1997.

112. P. Boulanger, J. Taylor, S. F. El-Hakim and M. Rioux. How to virtualize reality: an application to the re-creation of world heritage sites. In *Proceedings of the Conference on Virtual Systems and MultiMedia. Gifu, Japan*. NRC Technical Report 41599. 1998.

113. J.-A. Beraldin, F. Blais, L. Cournoyer, R. Rodella, F. Bernier and N. Harrison. Digital 3D imaging system for rapid response on remote sites. In *Proceeding of the 2nd International Conference on 3D Digital Imaging and Modeling*, Ottawa, Canada: pp. 34–45. October, 1999.

114. J. Y. Zheng and L. Z. Zhong. Virtual recovery of excavated relics. *IEEE Computer Graphics and Applications*, 19(3):6–11, May–June, 1999.

115. K. Ikeuchi *et al*. Modeling cultural heritage through observation. In *Proceedings of the First Pacific Rim Conference on Multimedia*. Sydney, Australia: December, 2000, (in press).

116. S. Rusinkiewicz and M. Levoy. A multiresolution point rendering system for large meshes. In *Proceedings of SIGGRAPH 00*, Computer Graphics Proceedings, Annual Conference Series. pp. 343–352. 2000.

117. F. Bernardini, I. Martin, J. Mittleman, H. Rushmeier and G. Taubin. Building a digital model of Michelangelo's Florentine Pietá. *IEEE Computer Graphics and Applications*, 22(1):59–67, 2002.

118. K. Robinette, H. Daanen and E. Paquet. The Caesar project: a 3D surface anthropometry survey. In *Proceedings of the 2nd International Conference on 3D Digital Imaging and Modeling*, Ottawa, Canada: pp. 380–387. October, 1999.

119. J. Yau. A texture mapping approach to 3D facial image synthesis. *Computer Graphics Forum*, 7(2):129–134, 1988.

120. M. Nahas, H. Hutric, M. Rioux and J. Domey. Facial image synthesis using skin texture recording. *The Visual Computer*, 6(6):337–343, 1990.

121. S. Marschner, S. Westin, E. Lafortune, K. Torrance and D. Greenberg. Image-based BRDF measurement including human skin. In *Proceedings of the 10th Eurographics Workshop on Rendering*, Granada, Spain: pp. 131–144. June, 1999.

122. S. Marschner, B. Guenter and S. Raghupathy. Modeling and rendering for realistic facial animation. In *Proceedings of the 11th Eurographics Workshop on Rendering*. Brno, Czech Republic: June, 2000.

123. P. Debevec, T. Hawkins, C. Tchou, H.-P. Duiker, W. Sarokin and M. Sagar. Acquiring the reflectance field of a human face. In *Proceedings of SIGGRAPH 00*, Computer Graphics Proceedings, Annual Conference Series. pp. 145–156. 2000.

124. A. Haro, B. Guenter and I. Essa. Real-time, photo-realistic, physically based rendering of fine scale human skin structure. In K. Myzkowski and S. Gortler (eds), *Rendering Techniques '01 (Proceedings of the 12th Eurographics Rendering Workshop)*. 2001.