Available online at www.sciencedirect.com
SCIENCE DIRECT®

Image and Vision Computing 22 (2004) 623–635

www.elsevier.com/locate/imavis

Automatic 3D model reconstruction based on novel pose estimation and integration techniques

Soon-Yong Park*, Murali Subbarao

Department of Electrical and Computer Engineering, State University of New York at Stony Brook, Stony Brook, NY 11794-2350, USA

Received 20 September 2002; received in revised form 22 December 2003; accepted 14 January 2004

Abstract

An automatic three-dimensional (3D) model reconstruction technique is presented to acquire complete and closed 3D models of real objects. The technique is based on novel approaches to pose estimation and integration. Two different poses of an object are used because a single pose often hides some surfaces from a range sensor. A second pose is used to expose such surfaces to the sensor. Two partial 3D models are reconstructed for two different poses of the object using a multi-view 3D modeling technique. The two 3D models are then registered in two steps—coarse registration, and its refinement. Coarse registration is facilitated by a novel pose estimation technique, which estimates a rigid transformation between two models. The pose is estimated by matching a stable tangent plane (STP) of each pose-model with the base tangent plane, which is invariant for a vision system. We employ geometric constraints to find the STP. After registration refinement, two models are integrated to a complete 3D model based on voxel classification defined in multi-view integration. Texture mapping is done to obtain a photo-realistic reconstruction of the object. Reconstruction results and error analysis are presented for several real objects.

Keywords: 3D reconstruction; Range image; Registration; Pose estimation; Pose integration

1. Introduction

Three-dimensional (3D) model reconstruction of real objects is a topic of much interest in *Computer Vision* and *Computer Graphics*. Its application areas range from *Virtual Reality* and *Computer Animation* to *E-Commerce*. One topic of research interest today in 3D model reconstruction is the acquisition of a complete 3D model from multiple views of an object. There have been two major approaches in this topic. The first approach is based on merging multiple range images (or 3D partial surfaces) into a complete 3D model [3,5,7,22]. Range images can be acquired by several techniques such as *Laser-ranging* or *Stereo Vision*. The other approach is based on processing photographic images using volumetric modeling techniques such as *Voxel Coloring* or *Shape-from-Silhouettes* [6,17,18].

In this paper, we present a photo-realistic 3D model reconstruction technique based on merging of multi-view (or *n*-view) range images of an object. Multi-view range images can be acquired by either moving a range sensor or a target object. A moving object on a turntable with a fixed sensor [5,6], or a moving sensor with a fixed object [1,17,25,26] has been used by researchers, in addition to other variations [8,13,24]. Our approach also employs a turntable stage to obtain multi-view images of the object. Calibration parameters of the turntable are used for coarse registration of multi-view range images.

Most investigations on 3D model reconstruction are limited to using a single pose of an object. For example, an object is placed on a turntable in a fixed pose, while a range sensor obtains multi-view images of the object. However, for many real objects, using a single pose yields only a partial 3D model because some surfaces of the object remain hidden from the range sensor for any given pose due to occlusion, concavities, etc. For example, a tea cup placed upright on a turntable hidden the bottom surface of the cup from the range sensor. A 3D model of such hidden surfaces could be reconstructed by placing the object in a different suitable pose (e.g. by placing the tea cup on its side) and sensing the visible shape. This yields a second partial 3D

* Corresponding author.
E-mail addresses: parksy@ece.sunysb.edu (S.-Y. Park); murali@ece.sunysb.edu (M. Subbarao).

0262-8856/$ - see front matter © 2004 Elsevier B.V. All rights reserved.
doi:10.1016/j.imavis.2004.01.002

model of the object for the new pose. In order to obtain a complete 3D model, the two 3D models reconstructed from the two different poses need to be registered and integrated. However, registration and integration of two partial 3D models is a difficult problem. For this reason, only a few researchers have considered this problem.

Allen and Yang [1] stitch a bottom surface of an object by matching edge features of the object's 3D model with an edge image of the bottom. They acquire multi-view range images of a fixed object using a moving range finder. After reconstructing a 3D model, they acquire a partial shape of the bottom surface and stitch its shape and texture to the 3D model using a feature matching technique. Wong and Cipolla [28] employ the shape-from-silhouettes technique for 3D model generation and combine a *structure-from-motion* technique to estimate registration parameters between multiple views. However, they manually register top and bottom surfaces of the object. Niem [17] also reconstructs complete 3D models using the shape-from-silhouettes technique, registers, and integrates top and bottom surfaces manually. Lensch et al. [14] and Iwakiri and Kaneko [10] use silhouette matching techniques to register and stitch an object's textures to its 3D model. Huber [8] also presents a 3D reconstruction technique using an unconstrained registration of multi-view partial shapes. He registers the partial shapes using *Spin* images and a graph searching technique. Rusinkiewicz et al. [24] use a structured light pattern and a real-time registration technique to merge partial 3D point sets in real-time.

A schematic diagram of our 3D modeling system is shown in Fig. 1. First, we generate two 3D models (let us call them *pose-models*) from two different poses of an object. Each 3D pose-model is reconstructed by a volumetric modeling technique, which is similar to Ref. [5]. We use turntable parameters to coarsely register n-view range images and refine them using a *point-to-plane* registration technique [2]. Marching cubes (MCs) algorithm is employed to polygonize volumetric space into triangle meshes [2,15].

Registration of two pose-models consists of two steps, coarse registration and its refinement. We use a novel pose estimation technique of the 3D pose-models to determine coarse parameters [20]. The pose estimation technique finds stable tangent planes (STPs) on a 3D model, which are candidates for the base tangent plane (BTP) of the other model and vice versa. When we place a rigid object on the flat top of a turntable, the object rests with its outer surface touching the table top. The planar table top will be a tangent plane of the object's surface. We call the planar table top the BTP. The BTP is invariant with respect to the object's pose and the world coordinate system. STPs are obtained on a pose-model using three geometric constraints, which are given in Section 3. We match the BTP of the first pose-model to a STP of the second pose-model, and simultaneously match the BTP of the second pose-model to a STP of the first pose-model. The best-matching planes from two models are used to estimate the transformation between them.

A novel pose integration technique is presented to merge two pose-models into an accurate and complete 3D model. The novelty of our technique is integration of two incomplete iso-surfaces, which represent the two pose-models. Since the two pose-models are available after integration of n-view images, we integrate two iso-surfaces rather than $2\times n$-view range images. The integration technique merges the pose-models by heuristically combining the signed distance and the class of a voxel from each pose. Error analysis using two ground truth objects is presented to show the geometric accuracy of our technique. 3D reconstruction of photo-realistic 3D models is presented on several real objects.

2. Reconstruction of a single pose-model

2.1. Range image acquisition

We use a stereo vision camera to obtain range images and a turntable to change viewing direction to an object.

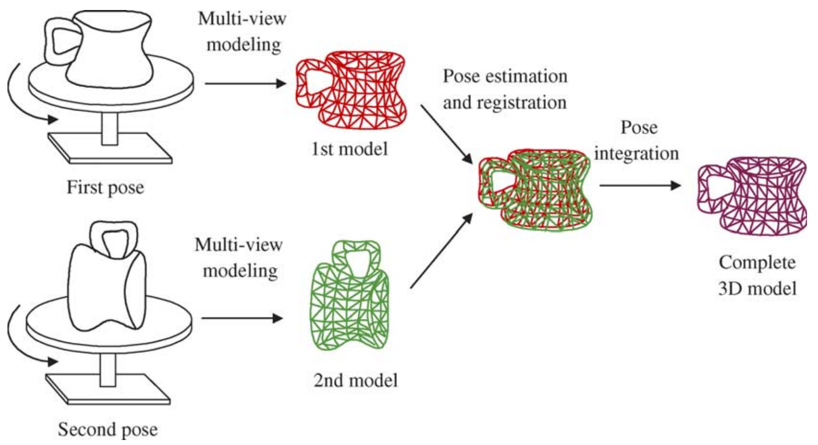


Fig. 1. Schematic diagram of our 3D modeling.

Both stereo camera and turntable are calibrated by the Tsai's calibration algorithm [27]. In order to introduce contrast on object's surface for stereo matching, we project a random dot pattern using a slide projector. For each pair of stereo image, we obtain a range image using a multi-resolution stereo matching technique based on a Gaussian Pyramid. A correlation based stereo matching is used at each level of the Gaussian pyramid. Normalized Gaussian low-pass filter with standard deviation of 1.0 is then applied to the range image to obtain sub-pixel accuracy. Object's multiple silhouettes are also segmented by a *blue screen* technique, and they are used later to carve out nonobject space in a volumetric integration step [19].

2.2. Multi-view registration

Calibration parameters of the vision system are used for coarse registration of multiple range images obtained from n-view directions. In order to refine the registration, a point-to-plane registration technique is employed to minimize transformation errors between all overlapping shapes [2,4]. Our approach is similar to Ref. [2]. To register a pair of overlapping range images, we search a triangle in the destination image, which is intersected by a vertex normal in the source image. The intersection point on the destination triangle is computed by interpolating three vertices of the triangle.

It is well known that registration errors accumulate if partial shapes are registered pairwise. Therefore, a multi-view registration technique, which evenly distributes the registration error in all overlapping regions, is needed. We set 0th view V_0 as the reference frame of multi-view registration. To register a partial shape S_i, where i is a view number, we search intersecting points only on the partial shapes S_{i-1} and S_{i+1}. This approach works well because a view point V_i has more overlapping shapes with its adjacent views V_{i-1} and V_{i+1} than others.

2.3. Multi-view integration

A volumetric integration technique is widely used for 3D reconstruction. Given multiple partial shapes of an object, we find the iso-surface of a 3D model in a grid of voxel space, and convert it to a mesh model using the MCs algorithm [15]. Because our vision system uses a stereo camera for range image acquisition, there are inherent mismatching errors on the range image. In order to accurately integrate partial shapes, we classify the voxel space into multiple regions based on signed distances $d_i(\mathbf{p})$, which is the distance from a voxel $\mathbf{p}$ to the ith range image $\mathcal{S}_i$. Let us assume there are N-view range images in total and $N_0(\leq N/2)$ views are overlapping each other. If there are at least two surfaces in $\mathcal{S}_i$, $i = 0, \ldots, N-1$, whose signed distance $d_i(\mathbf{p})$ from $\mathbf{p}$ is less than a threshold d_{TH}, then we consider they are overlapping surfaces. In this paper, we assume there are $N/2$ overlapping surfaces in maximum, which is 360°/2 in angle. In practice, the number of overlapping surfaces is determined by the distance and angle from a voxel to candidates of overlapping surfaces [19]. According to the sign of $d_i(\mathbf{p})$, we divide the grid of voxel space into four regions as follows (see Fig. 2):

- If at least two distances $|d_i(\mathbf{p})|$ are shorter than a threshold d_{TH}, where $i = 0, \ldots, N-1$, a voxel $\mathbf{p}$ is considered to be in an overlapping region and $\mathbf{p} \in P_{\mathrm{overlap}}$.
- If all $d_i(\mathbf{p})$ have positive sign and $|d_i(\mathbf{p})| > d_{\mathrm{TH}}$ for at least $N-1$ distances, the voxel is outside the object and $\mathbf{p} \in P_{\mathrm{outside}}$.
- If all $d_i(\mathbf{p})$ have negative sign and $|d_i(\mathbf{p})| > d_{\mathrm{TH}}$ for at least $N-1$ distances, the voxel is inside the object and $\mathbf{p} \in P_{\mathrm{inside}}$.
- Else, we assume that the voxel is in nonoverlapping area and $\mathbf{p} \in P_{\mathrm{nonoverlap}}$.

The threshold changes the smoothness of an integrated surface. We usually set d_{TH} as the double of the voxel size. For example, if the length of an edge of a voxel is d_x, then $d_{\mathrm{TH}} = 2d_x$. This assumption mostly gives reasonable results. The implicit distance $D(\mathbf{p})$ of the voxel $\mathbf{p}$ is a function of signed distances $d_i(\mathbf{p})$ to all overlapping shapes, i.e. $D(\mathbf{p}) = f(d_0(\mathbf{p}), \ldots, d_{N_0}(\mathbf{p}))$. For example, when a voxel

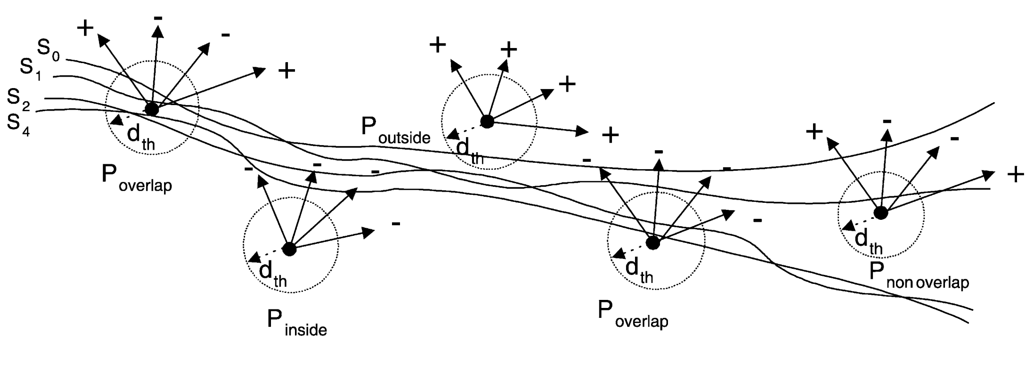


Fig. 2. Voxel classification based on signed distance.

is in P_{overlap}

$$D(\mathbf{p}) = \frac{\sum w_i(\mathbf{p}) d_i(\mathbf{p})}{\sum w_i(\mathbf{p})},$$

where $w_i(\mathbf{p})$ is the weight of the voxel to ith view [5]. However, when the voxel is in one of the other classes, $D(\mathbf{p})$ is computed in different ways based on the voxel's class. See Ref. [19] for more details.

3. Pose estimation and registration

We reconstruct two 3D pose-models of an object by employing the volumetric modeling technique described in Section 2. We place the object in an upright pose for the first pose-model and on its side for the second pose-model. This section presents registration of two 3D pose-models.

3.1. Base tangent plane

We employ a novel pose estimation technique based on geometric constraints of our vision system [20]. This technique finds a STP on a 3D model, which can be matched to the base tangent plane (BTP) of the other model. The BTP is a global tangent plane in the sense that it will not intersect the object's volume anywhere (in contrast, a local tangent plane may intersect the object's volume at a point far from the point of tangency). The BTP is invariant with respect to the object's pose and the world coordinate system. From the definition of the BTP, we find that there exists a unique tangent plane of the first pose-model, which corresponds to the BTP of the second pose, which is also a tangent plane of the second pose-model.

Suppose an object is placed on the turntable with two different poses, *Pose1* and *Pose2* as shown in Fig. 3. Then there is a unique tangent plane $\mathscr{T}_1$ (its normal is $\hat{n}_{T1}$) in the first pose which matches the BTP $\mathscr{B}(\hat{n}_B)$ in the second pose. Similarly, there is a unique plane $\mathscr{T}_2(\hat{n}_{T2})$ in Pose2, which matches $\mathscr{B}(\hat{n}_B)$ in Pose1. Because $\hat{n}_B$ is a common and invariant vector in this system, we can estimate a rotation matrix using $\hat{n}_{T1}$ and $\hat{n}_{T2}$.

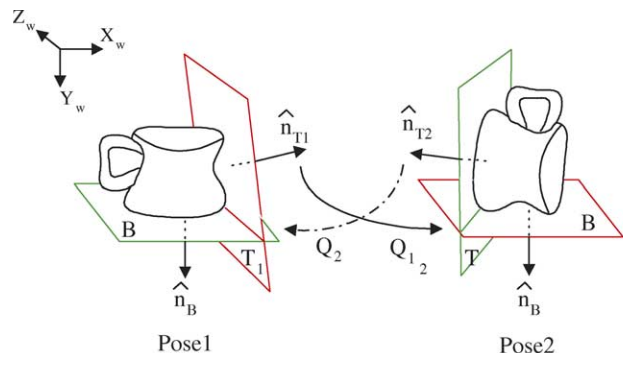


Fig. 3. A tangent plane of the first pose $\hat{n}_{T1}$ uniquely matches the BTP of the second pose $\hat{n}_B$, and vice versa.

Let $\mathbf{Q}_2$ be an initial transformation matrix which aligns $\hat{n}_{T2}$ with $\hat{n}_B$. After the alignment, two models are registered except one degree of freedom along the axis of $\hat{n}_B$. Then, by aligning the transformed vector $\mathbf{Q}_2\hat{n}_B$ with $\hat{n}_{T1}$, we estimate the transformation matrix $\mathbf{Q}_{21}$. Coarse translation between two models is estimated by center of mass (CoM) of the models. The translation and rotation parameters are later refined in a pose refinement step in Section 3.4. Estimating the translation using CoM is simple and accurate enough to refine the parameters of two pose-models.

We assign a new coordinate system for each tangent plane $\mathscr{T}_1$ and $\mathscr{T}_2$. Details on this will be described in Section 3.2. Consequently, our technique finds tangent planes $\mathscr{T}_1$ and $\mathscr{T}_2$ from two pose-models and the transformation matrix $\mathbf{Q}_{21}$, which minimizes a cost function between the two pose-models. The pose error is estimated in terms of sum of square difference (SSD) error between them. Suppose a vertex $\mathbf{p}_{1i}$ in Pose1 corresponds to another vertex $\mathbf{p}_{2i}$ in Pose2. In a point-to-tangent plane registration, for example, $\mathbf{p}_{2i}$ can be an intersection point on a destination surface and $\mathbf{p}_{1i}$ is its conjugate on a source surface. Therefore, pose error ϵ_p is measured by

$$\epsilon_p = \sum_{i=0}^{K} \|p_{1i} - \mathbf{Q}_{21} p_{2i}\|^2, \tag{1}$$

where K is the number of vertices in Pose1 or Pose2.

3.2. Stability constraints

The surface of a 3D model is represented by a finite number of triangles [15]. Therefore, there will be a finite number of tangent planes on each pose-model. Let a set of tangent planes in Pose1 be $\{\mathscr{T}_1\}$, and $\{\mathscr{T}_2\}$ be another set of tangent planes in Pose2. If we assign a tangent plane to every vertex, however, measuring the cost function for all combinations of two sets $\{\mathscr{T}_1\}$ and $\{\mathscr{T}_2\}$ is computationally expensive.

In order to reduce computational complexity, we remove local or unstable tangent planes by employing geometric constraints. A key idea for employing the constraints is that the object is placed on the turntable in a stable pose and the turntable is horizontal. Three constraints are as follows:

(1) *Base plane constraint*. An object is placed on the BTP, which is one of the global tangent planes of the object. This BTP does not intersect the object's volume.
(2) *Stability constraint*. The BTP of the turntable is horizontal and the object is in a stable pose. Therefore, the projection of the CoM to a STP is always inside the convex hull of its supporting vertices.
(3) *Height constraint*. If two pose-models are correctly registered, their heights will be very similar. (It may not be the same, because of noise).

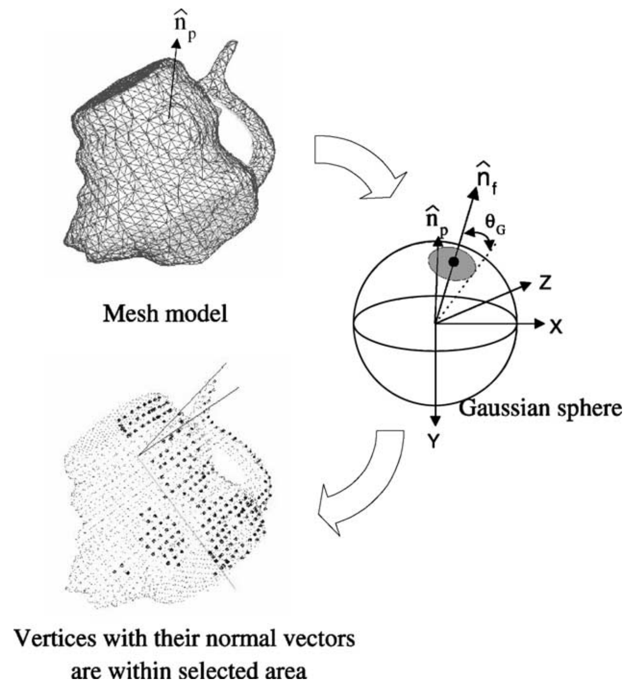


Fig. 4. Initializing tangent plane and its supporting vertices.

Based on the constraints above, we consider only STPs on the model. These constraints greatly reduce the number of tangent planes and the computation time of pose estimation.

3.3. Pose estimation

3.3.1. Finding tangent plane

As the first step of pose estimation, EGI (Extended Gaussion Image)s of two pose-models are constructed and all candidates of STPs on the tessellated Gaussian spheres are obtained [9,11]. Suppose we construct an EGI (Extended Gaussion Image) (Extended Gaussion Image) of a 3D mesh model of an object as shown in Fig. 4. The EGI (Extended Gaussion Image) consists of a set of tessellated polygons, where each polygon f is represented by its normal $\hat{n}_f$ and its area. Area of the polygon is the number of such triangles of the model that their normals are different in angle with respect to $\hat{n}_f$ by less than a tessellation angle [9,11]. Let $\hat{n}_T$ be the normal of a tangent plane $\mathscr{T}$, and $T(\mathbf{p}) = 0$ is the plane equation associated to $\mathscr{T}$, where $\mathbf{p} = (x_p, y_p, z_p)$ is a vertex on the plane. Then, we initialize the plane using $\hat{n}_f$ such that $T(\mathbf{p}) = \hat{n}_f \cdot \mathbf{p} + D_T$, where D_T is the distance from the origin to the plane.

Because we search for a STP which can support the object as a BTP, we update $T(\mathbf{p})$ by employing its supporting vertices. In an ideal case, all supporting vertices of $T(\mathbf{p})$ lie on the plane. Therefore, it is necessary that

$$T(\mathbf{p}_i) = 0 \tag{2}$$

and

$$\hat{n}_f \cdot \hat{n}_{p_i} = 1, \qquad \forall i = 0, \ldots, N_s - 1 \tag{3}$$

for the stability of the object [12], where N_s is the number of supporting vertices. In a real situation, however, supporting vertices are not always on the plane due to inherent noise on the surface of the model. Therefore, we need to find some vertices whose normal vectors $\hat{n}_p$ and their dot product, $\hat{n}_p \cdot \hat{n}_f$ are less than a threshold $\cos(\theta_G)$ as shown in Fig. 4. In this figure, the red-colored (dark-gray) vertices on the 3D model have their normals from $\hat{n}_f$ less than θ_G.

As a next step, we remove some of those vertices, which are far from the plane. Ideally, only one vertex on a tangent plane touches the surface of the model. However, by considering noise and stable computation, we search supporting vertices close to a reference vertex using the following technique. Let us define a vertex $\mathbf{p}_m$ whose projection distance to $\hat{n}_f$ is the maximum from the coordinate origin. Suppose we move the initial tangent

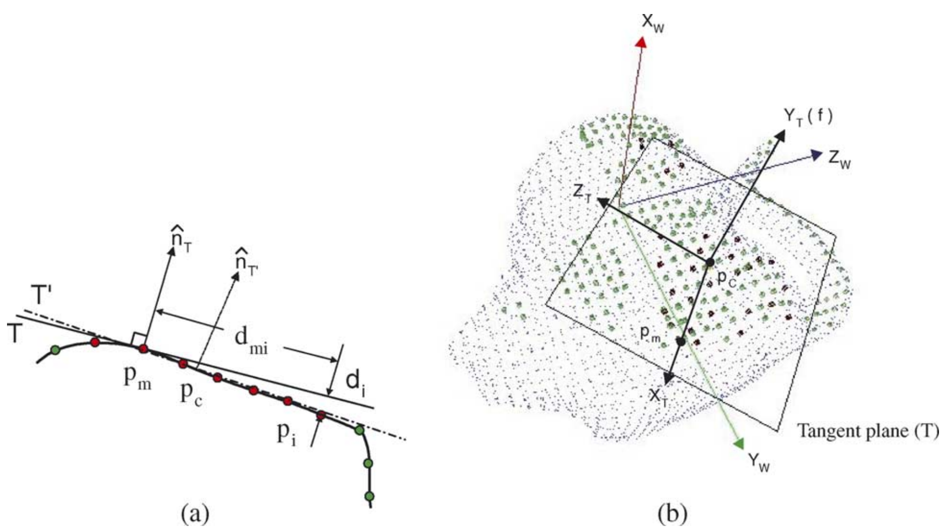


Fig. 5. Construction of a tangent plane on a 3D model. (a) Tangent plane $\mathscr{T}$ is updated by their supporting vertices. (b) Green (light gray) dots are all vertices with their normals within a threshold angle θ_G. However, only red dots (dark gray) consist of supporting points on the tangent plane.

plane so that $T(\mathbf{p}_m)$ touches $\mathbf{p}_m$. We call the vertex $\mathbf{p}_m$ the *frontier vertex* as shown in Fig. 5.

In Fig. 5(a), we select a vertex $\mathbf{p}_i$ as one of the supporting vertices, if its relative distance to the plane d_i/d_{mi} is less than a threshold. Here, d_i is the distance from $\mathbf{p}_i$ to the plane and d_{mi} is the distance from the frontier vertex to $\mathbf{p}_i$. In this paper, we fix the threshold of d_i/d_{mi} to 0.1. If the threshold increases, the more vertices will be involved to update the tangent plane. However, too many vertices can cause a deviation of $\hat{n}_T$ from the original orientation $\hat{n}_f$. After finding a set of supporting vertices $\mathscr{P}_s$, we move the origin of the plane to $\mathbf{p}_c$, the centroid of $\mathscr{P}_s$, and change the normal of $\hat{n}_T$ by averaging the normals of $\mathscr{P}_s$.

A new coordinate system is then generated for the tangent plane in order to obtain a transformation matrix to the reference coordinate system. Let $\mathbf{T}$ be the coordinate system associated to $\mathscr{T}$. We define the three axes of $\mathbf{T}$ as follows

$$Y_T = \hat{n}_T,$$

$$X_T = \overline{\mathbf{p}'_m - \mathbf{p}_c},$$

and

$$Z_T = X_T \times Y_T,$$

where $\mathbf{p}'_m$ is the projection of $\mathbf{p}_m$ to $\mathscr{T}$. In Fig. 5(b), the coordinate system of $\mathscr{T}$ is shown.

3.3.2. Finding stable tangent plane

A tangent plane $\mathscr{T}$ consists of supporting vertices and its own coordinate system. Let a set of tangent planes of Pose1 be $\{\mathscr{T}_1\}$, and $\{\mathscr{T}_2\}$ be the other tangent plane set of Pose2. To reduce computation complexity, we remove local or unstable tangent planes from further consideration by employing the three geometric constraints given in Section 3.2.

The first constraint is Base plane constraint. We check all vertices to determine if a tangent plane intersects the volume. If the dot product of any vertex $\mathbf{p}$ with the plane normal $\hat{n}_T$ is greater than the parameter D_T of the plane, we remove the plane. Due to noise on model's surface, we use a threshold $D_T + \delta_I$ instead of D_T.

Next constraint is Stability constraint. The BTP of the turntable is horizontal and the object is in a stable pose. Therefore, given the CoM of a 3D model, its projection to a STP, CoM_T is always inside the convex hull of the projections of all supporting vertices. The object will be unstable and fall over if CoM_T is outside the convex hull as shown in Fig. 6.

The last constraint is Height constraint. We reject any tangent plane when the height difference is greater than a threshold δ_H. Fig. 7 shows an example of removing inconsistent and unstable tangent planes. A 3D model is represented by a point clouds model and a tangent plane is represented by a square-shaped polygon. It shows that

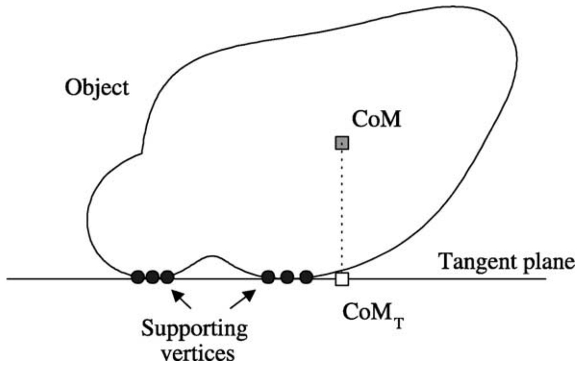


Fig. 6. An unstable tangent plane. The projection of CoM to the plane is outside the convex hull of supporting vertices.

the number of candidates for the matching tangent plane is greatly reduced by the three constraints.

3.3.3. Matching tangent planes

Rejection of unstable and local tangent planes significantly reduces the number of tangent planes. The last step in pose estimation is finding two matching tangent planes, one from each pose, which registers two 3D models with a minimum pose error. For every STP in $\{\mathscr{T}_1\}$, we derive a transformation matrix $\mathbf{Q}_{21}$ using every STP in $\{\mathscr{T}_2\}$, measure the pose error ϵ_p, and find two STPs which yield the best-matching. Let the coordinate of a STP in $\{\mathscr{T}_1\}$ be $\mathbf{T}_1$ and another STP in $\{\mathscr{T}_2\}$ be $\mathbf{T}_2$. Similarly, let the coordinate of the BTP be $\mathbf{B}$. The transformation matrix of

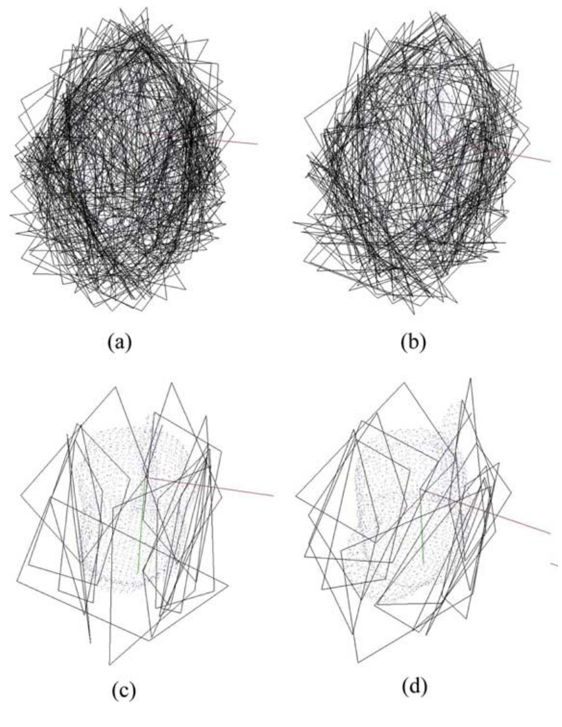


Fig. 7. Finding STPs on a 3D model based on geometric constraints. (a) Initial tangent planes (186 planes). (b) After removing volume-intersecting planes ($\delta_I = 3$ mm, 141 planes). (c) After removing unstable planes (18 planes). (d) After height comparison ($\delta_H = 3$ mm, 9 planes).

the second pose-model (a set of vertices $\mathscr{P}_2$) to the first pose-model (a set of vertices $\mathscr{P}_1$) is estimated by using $\mathbf{T}_2$ and $\mathbf{T}_1$. The transformation matrix first aligns $\mathbf{T}_2$ with the BTP is

$$\mathbf{B}' = \mathbf{T}_2^{-1}\mathbf{B}, \tag{4}$$

and then $\mathbf{T}_1$ with $\mathbf{B}'$ by rotating the model along the Y axis

$$\mathbf{T}_1 = \mathbf{R}_y^{-1}\mathbf{B}'. \tag{5}$$

The coordinate system of $\mathbf{B}$ is the same as the world (or common) coordinate system. A rotation matrix $\mathbf{R}_y$ aligns $\mathbf{B}'$ with $\mathbf{T}_1$. It is a rotation along the Y axis and computed by

$$\mathbf{R}_y = \begin{pmatrix} \cos\theta & 0 & -\sin\theta & 0 \\ 0 & 1 & 0 & 0 \\ \sin\theta & 0 & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

where

$$\begin{pmatrix} \cos\theta \\ \sin\theta \end{pmatrix} = \begin{pmatrix} B'_x + B'_z B'_z - B'_x \\ B'_x - B'_z B'_x + B'_z \end{pmatrix}^{-1} \begin{pmatrix} T_{1x} + T_{1z} \\ T_{1x} - T_{1z} \end{pmatrix}.$$

Let the translation from the origin of the common coordinate system to each CoM be $\mathbf{M}_1$ and $\mathbf{M}_2$. Then the pose transformation matrix $\mathbf{Q}_{21}$ from $\mathscr{P}_2$ to $\mathscr{P}_1$ is computed by

$$\mathscr{P}'_2 = \mathbf{M}_1\mathbf{R}_y^{-1}\mathbf{T}_2^{-1}\mathbf{M}_2^{-1}\mathscr{P}_2 \tag{6}$$

$$\mathscr{P}'_2 = \mathbf{Q}_{21}\mathscr{P}_2. \tag{7}$$

CoM of a 3D model is computed by a mass-computation technique [16]. The best-matching pose transformation $\mathbf{Q}_{21}$ is estimated by minimizing a cost function between two pose-models

$$\min_{\{\mathbf{T}_1 \in \{\mathscr{T}_1\}, \mathbf{T}_2 \in \{\mathscr{T}_2\}\}} \left\{ \sum \| \mathscr{P}_1 - \mathbf{Q}_{21}\mathscr{P}_2 \|^2 \right\}. \tag{8}$$

3.4. Pose registration

From the estimated pose $\mathbf{Q}_{21}$, we register and refine the second pose-model (the second set of multi-view range images) to the first pose-model. Refinement algorithm is based on a geometric registration technique and it is similar to that of multi-view registration [2]. Since registration among multi-view range images of each pose-model is already refined, we do the pose refinement only between two range image sets. From all partial surfaces in Pose2, we sample some of the vertices as control points, and find intersecting points from partial surfaces in Pose1. The first pose-model is fixed as a reference model and the second pose-model is registered iteratively until a pose error (translation and rotation between two control point sets) becomes close to zero.

4. Pose integration

After pose registration, two models are integrated into a complete 3D model. Pose integration computes final signed distance of the model $D_f(\mathbf{p})$, which is a function of signed distances to each pose

$$D_f(\mathbf{p}) = f(D_1(\mathbf{p}), D_2(\mathbf{p})), \tag{9}$$

where

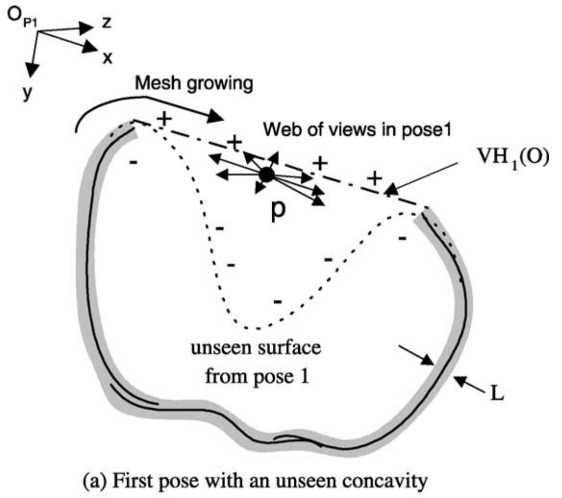
$$D_i(\mathbf{p}) = f(d_0^i(\mathbf{p}), \ldots, d_{N_0 i}^i(\mathbf{p})) \text{ for } i = 1, 2.$$

$D_i(\mathbf{p})$ is a weighted signed distance of a voxel $\mathbf{p}$ in pose i, $d_j^i(\mathbf{p})$ is the signed distance in pose i and view j, and N_0^i is the number of overlapping shapes in pose i. To derive the function $f(\,)$, we have to see which class $\mathbf{p}$ belongs to in each pose. If $\mathbf{p}$ is seen from any view in both poses, $D_f(\mathbf{p})$ can be computed as a weighted average. However, if there is any occlusion or concavity in either pose, $D_f(\mathbf{p})$ is estimated based on the class of $\mathbf{p}$ in both poses.

Let us consider situations that are more complex. Suppose there is a concavity on the object's surface as shown in Fig. 8. In Fig. 8(a) for example, $\mathbf{O}_{p1}$ is the common coordinate system of the first pose and all view points are nearly on the XZ plane of the coordinate system. We see that no view in the first pose can observe the concavity. But in Fig. 8(b), some of the views in the second pose can see the concavity. If a voxel $\mathbf{p}$ is inside of unseen surface region as shown in Fig. 8(a), it is classified as $\mathbf{p} \in P_{\text{inside}}$ in the first pose. Then, the MC algorithm closes a mesh model by following the visual hull $\text{VH}_1(O)$ of the multi-view frustum as shown in the figure. However, because it is seen from the second pose, there is no possibility that the MC algorithm marches on the same voxel $\mathbf{p}$. Instead, the algorithm must follow the object's surface, such as a voxel $\mathbf{p} \in P_{\text{overlap}}$ as in Fig. 8(b).

A technique of integrating two pose-models is based on the integration of two implicit models. If there are two implicit models $\mathbf{A}$ and $\mathbf{B}$, the merged model $\mathbf{C}$ can be represented simply as $\mathbf{C} = \mathbf{A} \cap \mathbf{B}$. In this case, a final signed distance $D_f(\mathbf{p})$ can be represented as $D_f(\mathbf{p}) = \max\{D_1(\mathbf{p}), D_2(\mathbf{p})\}$, where $D_i(\mathbf{p})$ is a signed distance of $\mathbf{p}$ in the pose i and $D_i(\mathbf{p}) < 0$ when $\mathbf{p}$ is inside an object [21, 23]. But, in a real system, selecting the maximum distance may shrink the volume of the final model. Rather than selecting the maximum, we average two weighted signed distances, when $\mathbf{p} \in P^i_{\text{overlap}}$ for $j = 1$ and 2. Otherwise, we heuristically select one of the distances or the maximum. Consequently, we select either the maximum $D_{\max}$, the average D_{avg}, or one of the distances $D_i(\mathbf{p})$ according to the results of multi-view integration as follows.

- $D_f(\mathbf{p}) = D_{\text{avg}}(\mathbf{p}) = \dfrac{\sum W_i(\mathbf{p}) D_i(\mathbf{p})}{\sum W_i(\mathbf{p})}$, *if* $\mathbf{p} \in P^1_{\text{overlap}}$ and $\mathbf{p} \in P^2_{\text{overlap}}$.



O_P1
z
x
y
Mesh growing
Web of views in pose1
VH_1(O)
p
unseen surface from pose 1
L
(a) First pose with an unseen concavity

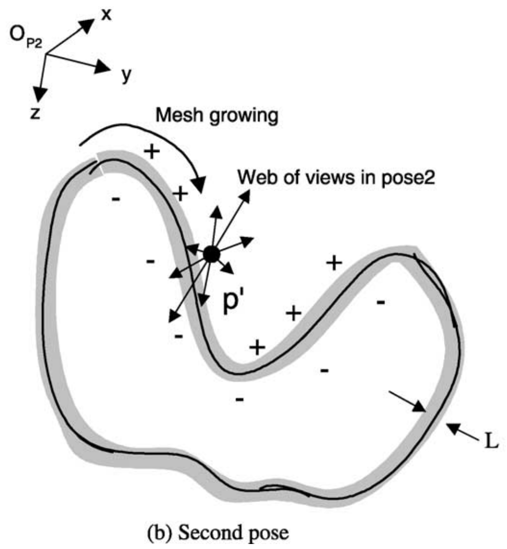


Fig. 8. Signed distance in a concave region. (a) MC algorithm follows the visual hull $\mathrm{VH}_1(O)$ to close the mesh model. (b) MC algorithm follows the actual surface of the object.

- $D_f(\mathbf{p}) = D_{\max} = \max\{D_1(\mathbf{p}), D_2(\mathbf{p})\}$, *if* $\mathbf{p} \in P^1_{\text{inside}}$ and $\mathbf{p} \in P^2_{\text{inside}}$.
- $D_f(\mathbf{p}) = D_1(\mathbf{p})$, if $\mathbf{p} \in P^2_{\text{inside}}$, or $D_f(\mathbf{p}) = D_2(\mathbf{p})$, if $\mathbf{p} \in P^1_{\text{inside}}$.

However, in some situations, a voxel will be in both $P^1_{\text{nonoverlap}}$ and $P^2_{\text{nonoverlap}}$. A voxel in a concave region sometimes belongs to this class. Consider an example shown in Fig. 9. In the figure, the signed distance from voxel p to two pose-models are $D_1(\mathbf{p})$ and $D_2(\mathbf{p})$ and $\mathbf{p} \in P^1_{\text{nonoverlap}}$ and $\mathbf{p} \in P^2_{\text{nonoverlap}}$. As shown in the figure, $D_1(\mathbf{p})$ has $(-)$ sign, even though the voxel is outside the object, because it is not visible from pose1. If $|D_1(\mathbf{p})| < |D_2(\mathbf{p})|$, the MC algorithm may choose the shorter one, then the voxel is considered inside the object. Typically, such errors happen near the visual hull and an example of some artifacts of an object with a concave region is shown in Fig. 10(a).

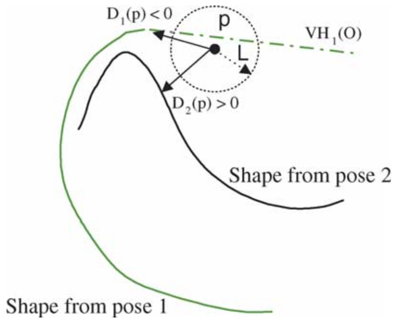


Fig. 9. Errors of mesh growing in a concave region. If $|D_1(\mathbf{p})| < |D_2(\mathbf{p})|$, voxel $\mathbf{p}$ has a wrong distance sign.

Rather than selecting the shorter of the two distances in the two pose-models, we compare visibility of the voxel for the views in each pose-model, i.e. check if the voxel is in $P_{\text{nonoverlap}}$ in both poses. If the voxel has many positive signs from the view points, it implies high visibility (our implicit representation assigns $(+)$ sign when a voxel is outside an object). The visibility of the voxel can be considered the number of positive distances to each pose. It can be easily determined by counting the number of positive distances of $d^i_j(\mathbf{p})$ in each pose. Therefore, we define more conditions for computing $D_f(\mathbf{p})$ in $P_{\text{nonoverlap}}$. If a voxel $\mathbf{p} \in P^1_{\text{nonoverlap}}$ and $\mathbf{p} \in P^2_{\text{nonoverlap}}$

- $D_f(\mathbf{p}) = D_1(\mathbf{p})$ if $\text{count}^+(d^1_j(\mathbf{p})) > \text{count}^+(d^2_j(\mathbf{p}))$.
- $D_f(\mathbf{p}) = D_2(\mathbf{p})$ if $\text{count}^+(d^2_j(\mathbf{p})) > \text{count}^+(d^1_j(\mathbf{p}))$.

where $\text{count}^+(\,)$ is the count of the number of positive distances in $d^i_j(\mathbf{p})$, where $j = 0, \ldots, N^i_0$. After applying these conditions for distance selection, we obtain an accurate reconstruction of the concave object's surface shown in Fig. 10(b).

5. Texture mapping

Texture mapping is the last step of our 3D reconstruction. After integrating two pose-models into a new 3D model, we

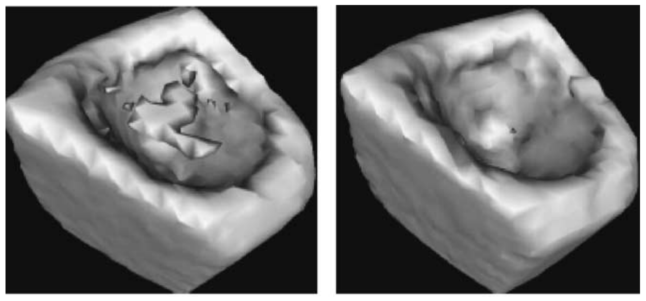


Fig. 10. Artifacts in a concave region. (a) Artifacts are near the visual hull. (b) Correct pose integration.

map textures on the surfaces of the model. We apply a general *view-independent texture mapping* technique. For each vertex on the object's surface, we find the best view point from all viewing directions of two poses. The best view is selected based on the cosine of the angle between the vertex normal and different viewing directions. If three vertices of a triangle face map to the same view point, we texture the triangle face using the image of that same view. If vertices on the triangle are mapped to different view points, we interpolate textures on the triangle using the barycentric coordinate system. We use range images of all viewing directions as *Z-buffers* to decide if a vertex is occluded from any view.

6. Experimental results

6.1. Error analysis

To analyze the accuracy of our 3D modeling technique, two ground truth objects are employed. Figs. 11 and 12 show the two ground truth objects and their dimension. The first object is a rectangular parallelepiped and the second one is a cylinder. Dimension of the reconstructed 3D models are measured and compared to those of the ground truth models.

3D ground truth models are generated by a computer simulation and represented by point clouds. Using an ICP (Itesative Closest Point)-based registration technique, we first register a ground truth model to its conjugate, which is reconstructed by the pose integration technique. Then root mean square (RMS) and maximum (MAX) dimensional errors are measured between all points on the reconstructed model and their closest conjugates on the ground truth. We iteratively register the 3D model to its ground truth until the registration errors (translation and rotation) converge close to zero. We use about 400 control points for 'Cubes' and about 300 points for 'Cylinder' objects. The registration results of two objects are 1.03 mm (Cubes) and 1.61 mm (Cylinder), respectively.

Table 1
RMS and maximum errors of 'Cubes'

Dimension	W (mm)	D (mm)	H (mm)	V (mm^3)
Size	60	60	90	324,000
RMS error	1.06	0.90	0.85	330,542
MAX error	2.99	3.03	2.17	
(%) error (RMS)	1.76	1.50	0.94	2.04

For the Cubes object, the RMS and MAX errors in W, H and D dimensions are measured for all vertices on corresponding planes—for example the top and bottom planes for H dimension—with respect to the closest vertices on the ground truth. Similarly for the Cylinder object, errors in R and H dimensions are measured using points on side surfaces, and top and bottom surfaces, respectively. Table 1 shows RMS and MAX errors in all dimensions of the Cubes object. We also measure a volumetric error (V) of the reconstructed model using a volume measuring technique described in Ref. [16]. Compared to the volume of the ground truth, the reconstructed model has 2.04% error as shown in the table. Table 2 shows the results of the Cylinder object.

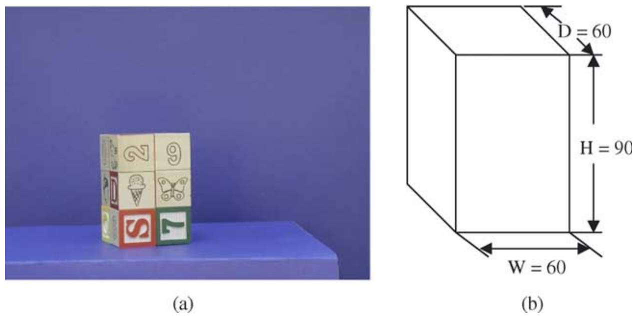


Fig. 11. (a) Object 'Cubes' used in error analysis. (b) Dimension in mm.

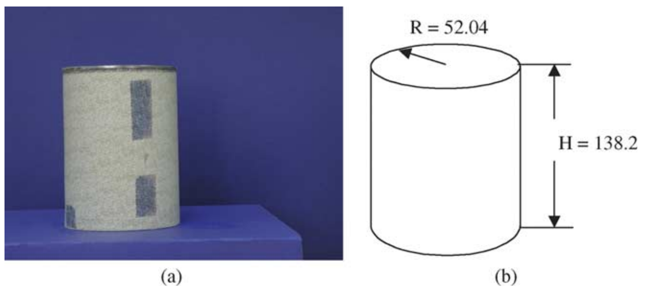


Fig. 12. (a) Object 'Cylinder' used in error analysis. (b) Dimension in mm.

6.2. Application to real objects

We have performed 3D modeling experiments on several real and complex objects. We use a Pentium III 1 GHz personal computer for the experiments. Each object is placed on a turntable and eight stereo image pairs are taken for each pose. A stereo image pair consists of two 1280×960 color images. Range image is reconstructed by a stereo matching technique with a resolution of 320×240. We place the object in a normal upright pose

Table 2
RMS and maximum errors of 'Cylinder'

Dimension	H (mm)	R (mm)	V (mm^3)
Size	138.2	52.04	1,175,797.4
RMS error	1.54	1.20	1,179,466.7
MAX error	4.56	4.42	
(%) error (RMS)	1.11	2.29	0.31

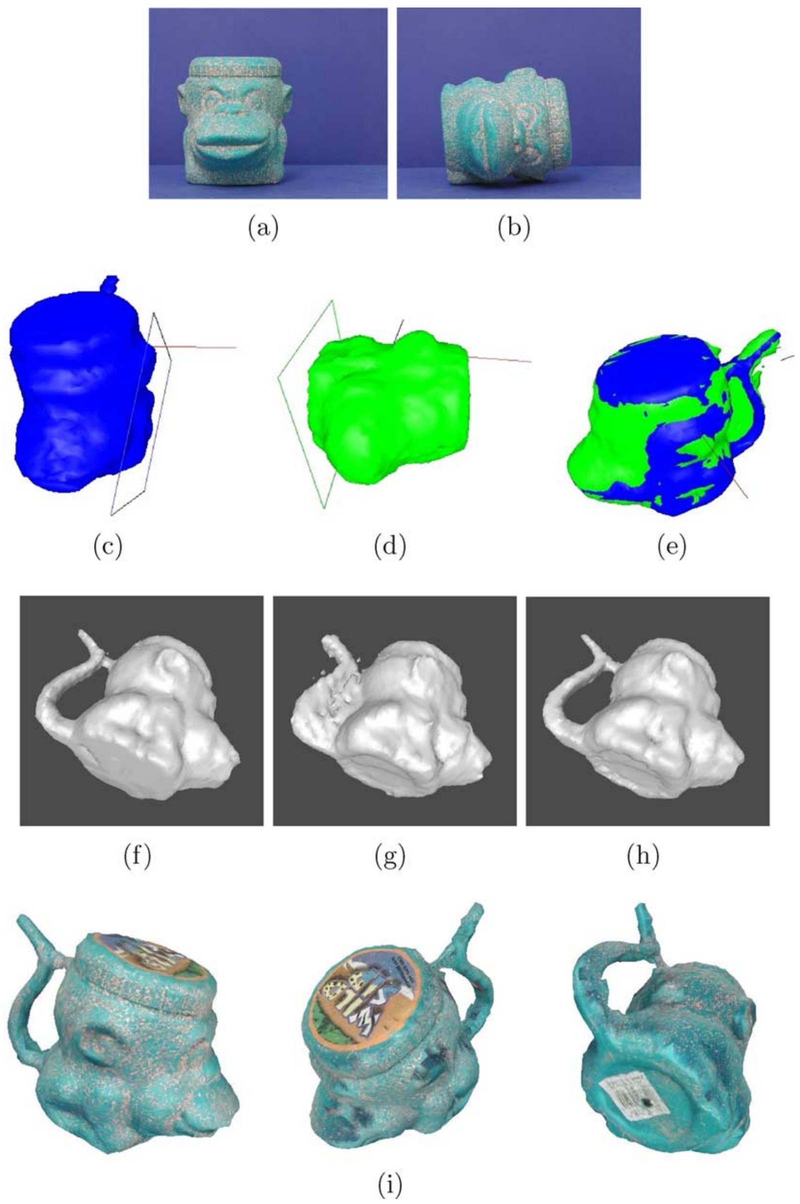


Fig. 13. Results of 'Monkey' (a) Pose1. (b) Pose2. (c) Pose1 mesh model and its matching tangent plane to Pose2. (d) Pose2 mesh model and its matching tangent plane to Pose1. (e) Coarsely registered models. (f) Pose1 mesh model. (g) Pose2 mesh model. (The bottom shape is accurately reconstructed, but some artifacts in occluded area). (h) Integrated model. (i) Novel views of the object.

for the first 3D pose-model, and on its side for the second 3D pose-model.

Fig. 13(a) and (b) shows images of two poses of a toy object 'Monkey'. We sprayed some color paint on its surface to introduce contrast. Reconstructed surface models are shown in Fig. 13(f) and (g), respectively. Pose between two pose-models is estimated and a matching STP on each pose-model is shown in Fig. 13(c) and (d). The squares represent the matching planes. Overlapping of two models after coarse registration is shown in Fig. 13(e). Fig. 13(h) is the integration result of two pose-models. The size of voxel used in this reconstruction is 4 mm. All surfaces of the object are reasonably reconstructed including the bottom

Table 3
Number of tangent planes

Object		Constraints			
		(a)	(b)	(c)	(d)
Monkey	Pose1	186	141	18	8
	Pose2	171	154	21	4
PolarBear	Pose1	180	150	18	7
	Pose2	167	144	11	4
Pokemon	Pose1	205	173	8	2
	Pose2	185	173	17	9

(a) Initial planes. (b) Intersection constraint. (c) Stability constraint. (d) Height constraint.

Table 4
Pose estimation error and estimation time ($D_i/D_{mi} = 0.1$)

Object	Monkey	PolarBear	Pokemon
Voxel size (mm)	4	4	2
$\cos(\theta_G)$	0.8	0.5	0.5
δ_I, δ_H (mm)	3.0	3.0	3.0
Avg. error (mm)	2.46	1.95	2.44
No. of vertices	3294	5106	6180
Time (s)	22.8	24.3	27.5

surface as shown in the figure. Fig. 13(i) shows three novel views of the integrated 3D model. The texture mapping results show accurate reconstruction on the object's surface including top and bottom.

The number of tangent planes at every rejection step in pose estimation is shown in Table 3. The Monkey object is polygonized with a voxel size of 4 mm. Table 4 shows threshold angle θ_G, threshold distances δ_H and δ_I, average pose error between two pose-models, number of vertices, and total estimation time. For measuring pose error, we use Eq. (1). From a vertex $\mathbf{p}_{1i}$ in $\mathcal{P}_1$, we find the closest vertex $\mathbf{p}_{2i}$ in $\mathcal{P}_2$ as the corresponding one. We sampled 50 vertices from each 3D model for computing the error measure.

The second object is a small toy, 'Pokemon'. Its height is only 8 cm. To introduce contrast on the object's surface, we project a random dot pattern using a slide projector. Fig. 14(c)–(e) shows pose estimation results. There are some occlusions in the second pose as shown in Fig. 14(g), but an integrated 3D model in Fig. 14(h)

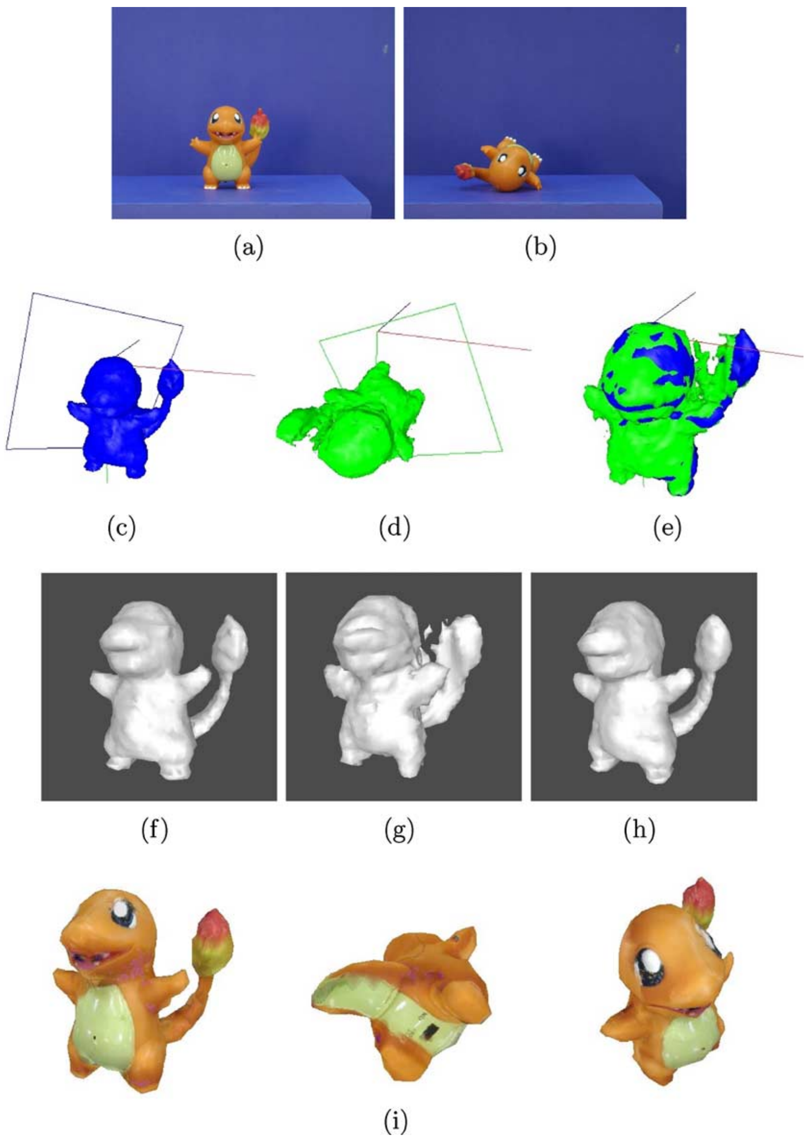


Fig. 14. Reconstruction of 'Pokemon'. (a) Pose1. (b) Pose2. (c) Pose1 mesh model and its matching tangent plane to Pose2. (d) Pose2 mesh model and its matching tangent plane to Pose1. (e) Coarsely registered models. (f) Pose1 mesh model. (g) Pose2 mesh model. (There are some artifacts in occluded area). (h) Integrated model. (i) Novel views of the object.

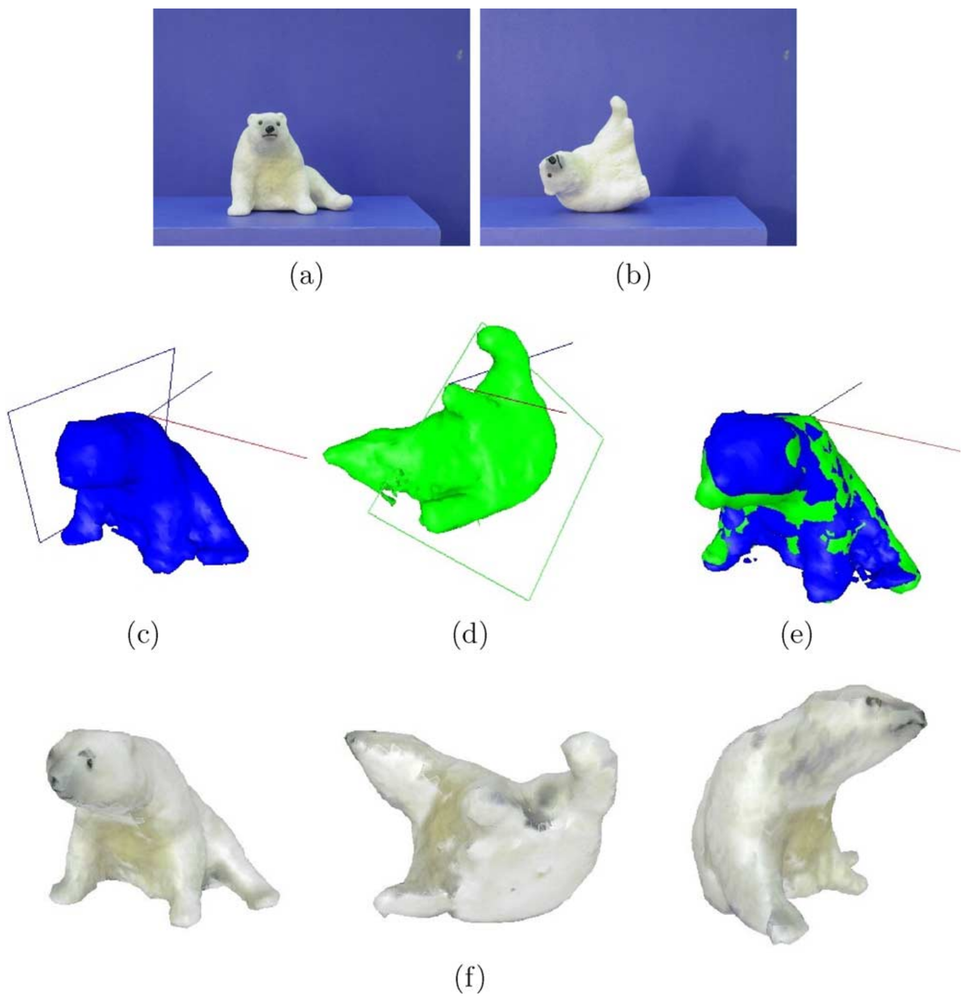


Fig. 15. Reconstruction of 'PolarBear'. (a) Pose1. (b) Pose2. (c) Pose1 mesh model and its matching tangent plane to Pose2. (d) Pose2 mesh model and its matching tangent plane to Pose1. (e) Coarsely registered models. (f) Novel views of the object.

shows an accurate reconstruction. Texture mapping results in Fig. 14(i) show novel views of the object. Results of the last object 'PolarBear' are in Fig. 15.

Reconstruction time depends on the number of vertices and triangles. When there are about 4000 vertices and 8000 triangles on a 3D model, approximate computation time is about 7 min for multi-view registration and integration. Pose estimation takes about 20–30 s as shown in Table 3. For integration of two pose-models, we need about 1 min for pose refinement, 2 min for pose integration, and 1 min for texture mapping.

7. Conclusions

A pose estimation and integration technique is presented for automatic and complete 3D model reconstruction. In order to avoid hidden surfaces that arise when only one pose is used, two poses are used. We introduce a novel technique for registering and integrating two partial 3D models for two different poses. The technique is based on matching STPs of one with the BTP of the other pose and vice versa. In the matching step, some geometric constraints are used to reduce computation and enhance accuracy. In the pose integration step, a voxel classification technique is used for accurate surface reconstruction. Texture mapped 3D models of several real objects are presented. Experimental results on real objects show that our novel pose registration and integration technique is effective.

References

[1] P. Allen, R. Yang, Registering, integrating, and building CAD models from range data, IEEE International Conference on Robotics and Automation (1998) 3115–3120.

[2] R. Bergevin, M. Soucy, H. Gagnon, D. Laurendeau, Toward a general multi-view registration technique, IEEE Transactions on Pattern Analysis and Machine Intelligence 18 (5) (1996).

[3] F. Bernadini, I.M. Martin, H. Rushmeier, High-quality texture reconstruction from multiple scans, IEEE Transactions on Visualization and Computer Graphics 7 (4) (2001) 318–332.

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

[5] B. Curless, M. Levoy, A volumetric method for building complex models from range images, in: Proceedings of SIGGRAPH (1996) 303–312.

[6] A.W. Fitzgibbon, G. Cross, A. Zisserman, Automatic 3D model construction for turn-table sequences, European Workshop SMILE'98, Lecture Notes in Computer Science 1506, 1998, pp. 155–170.

[7] A. Hilton, A.J. Stoddart, J. Illingworth, T. Windeatt, Reliable surface reconstruction from multiple range images, Proceedings of the ECCV'96, Springer-Verlag, Berlin, 1996, pp. 117–126.

[8] D.F. Huber, Automatic 3D modeling using range images obtained from unknown viewpoints, Proceedings of the Third International Conference on 3-D Digital Imaging and Modeling, IEEE Computer Society, Silver Spring, MD, 2001, pp. 153–160.

[9] K. Ikeuchi, M. Hebert, Spherical representations: from EGI (Extended Gaussion Image) to SAI, Technical Report CMU-CS-95-197, 1995.

[10] Y. Iwakiri, T. Kaneko, PC-based real-time texture painting on real world objects, Computer Graphics Forum 20 (3) (2001).

[11] S.B. Kang, K. Ikeuchi, 3D object pose determination using complex EGI (Extended Gaussion Image), Technical Report CMU-RI-TR-90-18, 1990.

[12] D.J. Kriegman, Computing stable poses of piecewise smooth objects, Image Understanding 55 (2) (1992) 109–118.

[13] P. Lander, A multi-camera method for 3D digitization of dynamic, real-world events, PhD Dissertation, Carnegie Mellon University, 1998.

[14] H. Lensch, W. Heidrich, H. Seidel, Automated texture registration and stitching for real world models, in: Proceedings of Pacific Graphics 2000, IEEE Computer Society Press, Silver Spring, MD, 2000, pp. 317–327.

[15] W.E. Lorensen, H.E. Cline, Marching cubes: a high resolution 3D surface construction algorithm, Computer Graphics 21 (4) (1987).

[16] B. Mirtich, Fast and accurate computation of polyhedral mass properties, Journal of Graphics Tools 1 (2) (1996) 31–50.

[17] W. Niem, Automatic reconstruction of 3D objects using a mobile camera, Image and Vision Computing 17 (1999) 125–134.

[18] S.M. Seitz, C.R. Dyer, Photorealistic scene reconstruction by voxel coloring, in: Proceedings of Computer Vision and Pattern Recognition Conference, 1997, pp. 1067–1073.

[19] S. Park, M. Subbarao, Automatic 3D model reconstruction using voxel coding and pose integration, International Conference on Image Processing, 2002.

[20] S. Park, M. Subbarao, Pose estimation of two-pose 3D models using the base tangent plane and stability constraints, Proceedings of Vision, Modeling, and Visualization, 2002.

[21] A. Pasko, V. Adzhiev, A. Sourin, V. Savchenko, Function representation in geometric modeling: concepts, implementation and applications, The Visual Computer 11 (8) (1995) 429–446.

[22] K. Pulli, Surface reconstruction and display from range and color data, PhD Dissertation, University of Washington, 1997.

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

[24] S. Rusinkiewicz, O. Hall-Holt, M. Levoy, Real-time 3D model acquisition, ACM Translations on Graphics (2002) (SIGGRAPH 2002).

[25] I. Stamos, P.K. Allen, Geometry and texture recovery of scenes of large scale, Computer Vision and Image Understanding 88 (2) (2002) 94–118.

[26] I. Stamos, M. Leordeanu, Automated feature-based range registration of urban scenes of large scale, Proceedings of Computer Vision and Pattern Recognition 2 (2003) 555–561.

[27] R.Y. Tsai, A versatile camera calibration technique for high-accuracy 3D machine vision metrology using off-the-shelf TV camera and lenses, IEEE Journal of Robotics and Automation 3 (4) (1987) 323–344.

[28] K. Wong, R. Cipolla, Structure and motion from silhouettes, in: Proceedings of Eighth IEEE International Conference on Computer Vision (ICCV01), Vancouver, Canada, vol. 2, 2001, pp. 217–222.