1 Introduction

Solid modeling is a useful tool for tasks such as representing the virtual environment for virtual reality systems; representing the real environment for robot programming; and modeling real objects for object recognition. Currently, most object models are constructed by human operators [2]. It would be much better to have a system that can automatically build models of real objects that it observes. If we can develop a reliable technique to generate accurate 3D object models by observing real objects from multiple views, we can reduce the effort and cost of model construction, and we can significantly broaden the application areas of solid modeling.

Observation-based modeling systems usually work with a sequence of images of the object(s), where the sequence spans a smoothly varying change in the positions of the sensor and/or object(s). Most previous systems have attempted to apply inter-frame motion estimates to successive pairs of views in a sequential manner [12]. Whenever a new view is introduced, it is matched with the previous view, and the transformation between these two successive views has to be recovered before the object model is updated. This sequential method does not work well in practice because local motion estimates are subject to noise and missing data. Local mismatching errors accumulate and propagate along the sequence, yielding erroneous object models.

Rather than sequentially integrating successive pairs of view, we should instead search for the statistically optimal object model that is most consistent with all the views. Although every single view provides only partial information of the object, it is likely that any part of the object will be observed a number of times along the sequence. Object modeling from this sequence of views can be formulated as an overdetermined minimization problem because significant redundancy exists among all the views.

1.1 Past work

Much work has been done in object modeling from a sequence of *range* images [4]. Most work assumed that transformation between successive views is either known or can be recovered, so that all data can be transformed to a fixed coordinate system. For example, Bhanu [3] rotated the object through known angles. Ahuja and Veenstra [1] constructed an octree object model from orthogonal views. Soucy and Laurendeau [16] proposed to triangulate each view and merge multiple views via a Venn diagram when the transformation is known. Because building a Venn diagram is combinatorial, only four-view merging is presented in their work. By finding the correspondences from intensity patterns in all eight views, Vemuri and Aggarwal [21] derived the inter-frame motion and transformed all eight range images to the first frame. Ferrie and Levine [9] merged multiple views using correspondence points which are identified by correlation over the differential properties of the surface. Parvin and Medioni [12] proposed to construct boundary representation (B-rep) object models from unregistered multiple range images. Each view of the object is represented as an adjacency graph where nodes represent surface patches with attributes and arcs representing adjacency between surfaces. To merge any two views, a rigid transformation has to be computed accurately. Most of previous approaches to modeling from a sequence of views are sequential. Thus, transformation errors accumulate and propagate from one matching to another, which may result in imprecise object models.

Inferring scene geometry and camera motion from a sequence of *intensity* image is also possible in principle. For example, Tomasi and Kanade [19] proposed a factorization method to simultaneously solve shape and motion under orthography, and Poelman and Kanade [13] extend it to the case of paraperspective projection. Szeliski and Kang [18] proposed a nonlinear optimization method to solve shape and motion under perspective. However, in [19][13], the task is formulated as a least squares problem where missing data due to occlusion and mismatching is extrapolated from measured data and estimated motion. Although three views of four points is theoretically sufficient in determining structure and motion [20], it is difficult in practice to find a good submatrix to do "row-wise" and "column-wise" extrapolation. Szeliski and Kang [18] proposed to assign a weight to each measurement and incorporated an object-oriented perspective projection in a nonlinear least squares problem. The very nature of the nonlinear least squares formulation requires standard techniques in nonlinear optimization, e.g., Levenberg-Marquardt, in which convergence to a local minimum may be a problem. In addition, most existing algorithms seem to be more useful for determining camera motion than for building 3D object models because the recovered object shape is defined by a collection of 3D points whose connectivity is not explicitly known.

The factorization method [5][13][19], in essence, is principal component analysis of some measurement matrix. Principal component analysis expresses the variance of the measurement matrix in a compact and robust way and has been extensively studied in computational statistics [7]. The singular value decomposition (SVD) method [10] is a straightforward solution when the measurement matrix is complete. When data is incomplete or missing, as often the case in practice, principal component analysis becomes much more complicated. Ruhe [15] first proposed a Gauss-Newton algorithm to solve this problem, taking advantage of the sparse, structured derivatives of the object function. Wiberg [22] generalized Ruhe's work to the cases where the rank of the measurement matrix is known. However, Wiberg's algorithm requires solving large pseudo-inverse matrices.

1.2 Our approach to multiple view merging

We propose to build object models from a sequence of range images. Our approach is to recover bounding surfaces and transformations simultaneously by employing principal component analysis with missing data. After segmenting all range images and establishing correspondences among different views, a composite graph is built. The object surface description and transformations among different views are recovered by solving a combination of two weighted least squares (WLS) problems.

There are two key observations to our approach of object modeling from a sequence of views. First, because of the redundancy in the sequence of images, we can get a reliable solution from an overconstrained minimization problem even when data is missing. Because only part of the object is visible in each view we cannot find correspondences among all surfaces between two views. Therefore, this is not a least squares (LS) problem but a WLS one where the weights are zeros for invisible regions. The difficulty is how to formulate the WLS problem properly and how to solve this problem without resorting to extrapolation of the unknowns. We present an algorithm to iteratively update the surface description and transformation so that the weighted least squares error is minimized.

The second observation lies in the first WLS problem of recovering surface normals and rotation matrices. The modeling problem can be decomposed into two smaller problems because recovering rotation is independent of translation. If we directly apply the WLS algorithm, we have to explicitly update nine parameters of every rotation matrix. It is well-known that the rotation matrix is a nonlinear function of only three independent parameters. Therefore, updating nine parameters (even with proper normalization afterwards) is not the best way to solve this problem. We solve this problem by representing rotations using quaternions.

To make an object model from a recovered set of surface equations, spatial connectivity among surfaces has also to be recovered. Spatial connectivity refers to the spatial relationship among surfaces, i.e., for each surface, what other surfaces it is connected to. The problem of surface connectivity is reduced to one of connectivity of supporting lines of a simple polygon, solved by a modified Jarvis' march algorithm that combines information on both algebraic level and signal level.

1.3 Organization of paper

In section 2 we discuss principal component analysis when data is missing. From a motivational example of modeling 12-faced polyhedra from a sequence of views, we formulate the multiple view merging as a problem of principal component analysis with missing data (PCAMD). Then we outline Wiberg's formulation of PCAMD, and modify the formulation by proper indexing of the objective function. The modified formulation is then generalized to a WLS minimization problem. An efficient PCAMD algorithm is presented to solve this WLS problem. In section 3 we formulate modeling the object and recovering transformations as a combination of two WLS problems. We compute the surface description and transformation by extracting the principal components of two highly rank-deficient measurement matrices with many missing elements, each of which forms a WLS problem. The first WLS problem of recovering rotation matrices and surface normals is further simplified using the quaternion representation of rotation. A two-step algorithm is presented to model the object from a sequence of segmented range images. Section 4 gives a brief description of a surface patch tracking system. Different modules in the tracking system, range image segmentation, adjacency graph building, and two-view merging are presented. In section 5 we show that the problem of surface connectivity can be reduced to one of connectivity of supporting lines of a simple polygon. Since the problem of establishing connectivity of supporting lines can be regarded as both a modified convex hull-like problem and a cell decomposition problem, we propose a modified Jarvis' march algorithm which successfully reconstruct the simple polygon. We demonstrate applicability and robustness of proposed PCAMD method by applying our approach to synthetic data and real range images in Section 6. We show that, given correspondence, four views are necessary to recover the shape of an arbitrary dodecahedron (12-face polyhedra). A study on synthetic data of a dodecahedron shows that our approach is robust with respect to noise and surface mismatching. Our method yields a statistically optimal model for a given set of views; this method improves as more views are incorporated. From a sequence of real range images, a polyhedral object model is precisely recovered using the proposed method. A complex toy house model is also reconstructed from a sequence of range images. Final comments and conclusions are presented in Section 7.

2 Principal Component Analysis with Missing Data

2.1 Motivational example

Suppose that our task is to make a model for a dodecahedron (*12*-faced polyhedra) from a sequence of segmented range images. A dodecahedron is a simple and interesting Platonic solid. Assume that we have tracked *12* faces over *4* nonsingular views. The segmented range images provide trajectories of plane coordinates $\{\mathbf{p}_p^{(f)} \mid f=1,\ldots,4,\ p=1,\ldots,12\}$, where $\mathbf{p} = (\mathbf{v}^T, d)^T$ represents a planar equation with surface normal $\mathbf{v}$ and normal distance to the origin d. Then we may form a 16×12 measurement matrix as follows:

$$W = \begin{bmatrix} \mathbf{p}_1^{(1)} & \mathbf{p}_2^{(1)} & \mathbf{p}_3^{(1)} & \mathbf{p}_4^{(1)} & \mathbf{p}_5^{(1)} & \mathbf{p}_6^{(1)} & * & * & * & * & * & * \\ \mathbf{p}_1^{(2)} & \mathbf{p}_2^{(2)} & \mathbf{p}_3^{(2)} & \mathbf{p}_4^{(2)} & * & * & \mathbf{p}_7^{(2)} & \mathbf{p}_8^{(2)} & * & * & * & * \\ \mathbf{p}_1^{(3)} & \mathbf{p}_2^{(3)} & * & * & * & \mathbf{p}_6^{(3)} & * & \mathbf{p}_8^{(3)} & \mathbf{p}_9^{(3)} & \mathbf{p}_{10}^{(3)} & * & * \\ * & * & * & * & * & * & \mathbf{p}_7^{(4)} & \mathbf{p}_8^{(4)} & \mathbf{p}_9^{(4)} & \mathbf{p}_{10}^{(4)} & \mathbf{p}_{11}^{(4)} & \mathbf{p}_{12}^{(4)} \end{bmatrix}$$

where every * indicates an unobservable face since there are only six visible faces from each nonsingular view. Our modeling task is now to recover the poses of all the *12* faces in a fixed coordinate system.

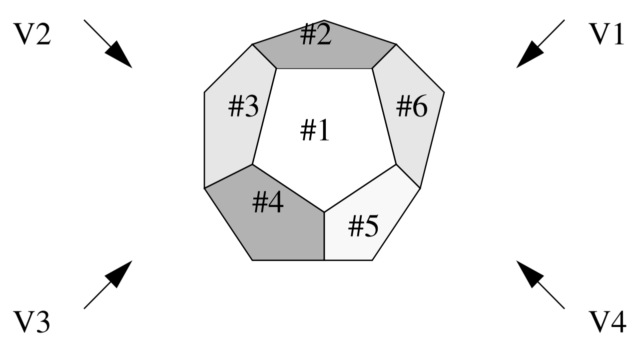


Figure 1 **Distinct views of a dodecahedron**

If the measurement matrix were complete, our task would be to average all those *12* faces over *4* views assuming data is noisy. In the absence of noise, any set of *12* faces from one of *4* views will do. The standard way to solve this problem is to apply SVD to this measurement matrix, whose rank is at most *4* (see section 3.1 for the argument). The measurement matrix can subsequently be factorized, with proper normalization, into a left matrix Q of transformation parameters and a right matrix P of plane coordinates

$$W = Q\ P$$

where

$$P = \begin{bmatrix}\mathbf{p}_1\ \mathbf{p}_2\ \dots\ \mathbf{p}_{12}\end{bmatrix},\ Q = \begin{bmatrix}Q^{(1)}\\ Q^{(2)}\\ Q^{(3)}\\ Q^{(4)}\end{bmatrix}$$

and $Q^{(f)}$ is the transformation of fth view with respect to the fixed world coordinate system, and $\mathbf{p}_p$ is the pth plane equation in the same world coordinate system. Singular value decomposition has also been successfully applied to shape and motion recovery from a sequence of intensity images [19].

Unfortunately, the measurement matrix is often incomplete in practice; it is not unusual for a large portion of the matrix to be unobservable. As we have seen in the above example, half of the measurement matrix is unknown. When the percentage of missing data is very small, it is possible to replace the missing elements with the mean or an extreme value; this is a common strategy in multivariate statistics [7]. However, such an approach is no longer valid when a significant portion of the measurement matrix is unknown.

One common practice in modeling from a sequence of images is to use extrapolation. For example, we can recover the transformation between view *1* and view *2* if there are at least three matched planar surfaces that are non-parallel [8]. Then we extrapolate the invisible planar surfaces in view *1* from its corresponding surfaces in view *2* which are visible using the transformation recovered. And apply the same extrapolation to invisible surfaces in view *1*. By repeating this process we can in principle extrapolate the locations of all invisible surfaces from visible surfaces [12]. A final step could be added to fine-tune the result by factorizing the extrapolated measurement matrix using SVD. A similar extrapolation approach "propagation method" [19] is used in motion and shape recovery from a sequence of intensity images.

The major problem with the extrapolation method is that once the estimated transformation is incorrect at any step, the extrapolated results will be erroneous. In sequential modeling errors accumulate and propagate all the way. The fine-tuning process at the last step would not improve the result dramatically since the extrapolated measurement matrix is inaccurate. To obviate this problem, we make use of more rigorous mathematical tools developed in computational statistics that caters for missing data without resorting to error sensitive extrapolation. We will demonstrate the formulation in this section and apply it to multiple view merging in the next section.

2.2 Wiberg's formulation

The problem of object modeling from a sequence of views shown in the previous section can be formulated as a problem of principal component analysis with missing data (PCAMD), which has been extensively studied in computational statistics. Ruhe [15] proposed a minimization method to analyze one component model when observations are missing. One component model decomposes an $F \times P$ measurement matrix into an $F \times 1$ left matrix and a $1 \times P$ right matrix. Wiberg [22] extended Ruhe's method to the more general case of arbitrary component model. We first outline Wiberg's formulation of principal component analysis with missing data, before proposing a modified formulation by appropriate indexing, and generalizing the problem as a WLS problem.

Suppose that an $F \times P$ measurement matrix W consists of P individuals from an F-variate normal distribution with mean $\bar{\mu}$ and covariance Σ. Let the rank of W be r. If the data is complete and the measurement matrix filled, the problem of principal component analysis is to determine $\tilde{U}$, $\tilde{S}$, and $\tilde{V}$ such that

$$\left\| W - \mathbf{e}\mu^T - \tilde{U}\tilde{S}\tilde{V}^T \right\|$$

is minimized, where $\tilde{U}$, and $\tilde{V}$ are $F \times r$ and $P \times r$ matrices with orthogonal columns, $\tilde{S} = diag(\sigma_i)$ is an $r \times r$ diagonal matrix, μ is the maximum likelihood approximation of the mean vector and $\mathbf{e}^T = (1, \ldots, 1)$ is an F-tuple vector with all ones. The solution to this problem is essentially SVD of the centered (or registered) data matrix $W - e\mu^T$.

If data is incomplete, we have the following minimization problem:

$$min \quad \phi = \frac{1}{2}\sum_{I} (W_{f,p} - \mu_p - \mathbf{u}_{f.}^T \mathbf{v}_{p.})^2 \qquad \text{(EQ 1)}$$

$$I = \{ (f, p) : \; W_{f,p} \; is\ observed \}$$

where $\mathbf{u}_{f.}$ and $\mathbf{v}_{p.}$ are column vector notations defined by

$$\begin{bmatrix} \mathbf{u}_{1.}^T \\ \cdots \\ \mathbf{u}_{F.}^T \end{bmatrix} = \tilde{U}\tilde{S}^{\frac{1}{2}} \qquad \text{(EQ 2)}$$

and

$$\begin{bmatrix} \mathbf{v}_{1.}^T \\ \cdots \\ \mathbf{v}_{P.}^T \end{bmatrix} = \tilde{V}\tilde{S}^{\frac{1}{2}} \qquad \text{(EQ 3)}$$

Lemma 1

A necessary condition to uniquely solve this problem (EQ 1) is $m \geq r\,(F+P-r) + P$ **where m is the number of observable elements in** W**.**

Proof:

It is trivially true that there are at most $r\,(F+P-r)$ independent elements from LU decomposition of an $F \times P$ matrix of rank *r*. Hence, to uniquely solve (EQ 1), the number of equations (*m*) has to be no fewer than the number of unknowns (*r(F+P-r)+P)*). (Q.E.D.)

To sufficiently determine the problem (EQ 1) more constraints are needed to normalize either the left matrix $\tilde{U}$ or the right matrix $\tilde{V}$.

If we write the measurement matrix W as an *m*-dimensional vector $\mathbf{w}$, the minimization problem can be written as

$$min \quad \phi = \frac{1}{2}\mathbf{f}^T\mathbf{f} \qquad \text{(EQ 4)}$$

where

$$\mathbf{f} = \mathbf{w} - \hat{\mu} - F\mathbf{u} = \mathbf{w} - G\tilde{\mathbf{v}} \qquad \text{(EQ 5)}$$

and

$$\mathbf{u} = \begin{bmatrix} \mathbf{u}_{1.} \\ \dots \\ \mathbf{u}_{F.} \end{bmatrix}, \tilde{\mathbf{v}} = \begin{bmatrix} \tilde{\mathbf{v}}_{1.} \\ \dots \\ \tilde{\mathbf{v}}_{P.} \end{bmatrix}, \tilde{\mathbf{v}}_{p.} = \begin{bmatrix} \mathbf{v}_{p.} & \mu_p \end{bmatrix}, \hat{\mu}_i = \mu_{p(i)}.$$

F and G are of dimension $m \times rF$ and $m \times (r+1)P$ respectively, and are computed by expanding every element f_i of $\mathbf{f}$

$$f_i = w_i - \mu_{p(i)} - u^T_{f(i).} v_{p(i).}$$

where the ith component of $\mathbf{w}$ is indexed to the *(f(i), p(i))*-th component of *W*, i.e., $w_i = W_{f(i), p(i)}$, and $W_{f,p} = w_{i(f,p)}$.

To solve the minimization problem stated by (EQ 4), the derivative of the objective function (with respect to $\mathbf{u}$ and $\tilde{\mathbf{v}}$) should be zero, i.e.,

$$\dot{\phi} = \begin{bmatrix} F^T F\mathbf{u} - F^T(\mathbf{w} - \hat{\mu}) \\ G^T G\tilde{\mathbf{v}} - G^T\mathbf{w} \end{bmatrix} = \mathbf{0} \qquad \text{(EQ 6)}$$

Obviously (EQ 6) is nonlinear because F is a function of $\mathbf{v}$ and G is a function of $\mathbf{u}$. In theory, any appropriate nonlinear optimization method can be applied to solve it. However, the dimensionality is so high in practice that we have to adapt the algorithm to make use of the special structure of the problem. It can been observed that:

(1) For fixed **u**, we have a linear least squares problem of **v**; for fixed **v**, we have a linear least problem of **u**;

(2) since (EQ 6) is also a bilinear problem of **u** and **v**, we can successively improve their estimates by using the updating technique in the NIPALS algorithm [15], i.e., for a given **v**, **u** is updated $\mathbf{u} = F^+(\mathbf{w} - \hat{\mu})$; for a given **u**, **v** is updated $\tilde{\mathbf{v}} = G^+\mathbf{w}$. F^+ and G^+ are the pseudo-inverses of F and G, respectively.

2.3 Modified Wiberg's formulation

In practice F and G are usually sparse matrices with many zeros. If we appropriately index $\mathbf{w}$ as $\mathbf{w}_1$, such that

$$\mathbf{f}_1 = \mathbf{w}_1 - \hat{\mu} - H\mathbf{u} \tag{EQ 7}$$

or

$$\mathbf{f}_1 = \begin{bmatrix} W_{1,1} \\ W_{1,2} \\ \cdots \\ W_{1,P} \\ \cdots \\ W_{F,1} \\ \cdots \\ W_{F,P} \end{bmatrix} - \begin{bmatrix} \mu_1 \\ \mu_2 \\ \cdots \\ \mu_P \\ \cdots \\ \mu_1 \\ \cdots \\ \mu_P \end{bmatrix} - \begin{bmatrix} V_{1,1} & \cdots & V_{r,1} & & & & \\ V_{1,2} & \ddots & & & & \mathbf{0}_{P\times r} & \\ V_{1,P} & & V_{r,P} & & & & \\ & & & \ddots & & & \\ & & & & V_{1,1} & & V_{r,1} \\ & \mathbf{0}_{P\times r} & & & & \ddots & \\ & & & & V_{1,P} & & V_{r,P} \end{bmatrix} \begin{bmatrix} U_{1,1} \\ U_{1,2} \\ \cdots \\ U_{1,r} \\ \cdots \\ U_{F,1} \\ \cdots \\ U_{F,P} \end{bmatrix} \tag{EQ 8}$$

and similarly, if we index $\mathbf{w}$ as $\mathbf{w}_2$, such that

$$\mathbf{f}_2 = \mathbf{w}_2 - K\tilde{\mathbf{v}} \tag{EQ 9}$$

or,

$$\mathbf{f}_2 = \begin{bmatrix} W_{1,1} \\ W_{2,1} \\ \cdots \\ W_{F,1} \\ \cdots \\ W_{1,P} \\ \cdots \\ W_{F,P} \end{bmatrix} - \begin{bmatrix} U_{1,1} & \cdots & U_{1,r} & 1 & & & & \\ U_{2,1} & \ddots & U_{2,r} & 1 & & \mathbf{0}_{F\times(r+1)} & & \\ & & & \cdots & & & & \\ U_{F,1} & & U_{F,r} & 1 & & & & \\ & & & & \ddots & & & \\ & & & & U_{1,1} & & U_{1,r} & 1 \\ & \mathbf{0}_{F\times(r+1)} & & & & & & \cdots \\ & & & & U_{F,1} & \ddots & U_{F,r} & 1 \end{bmatrix} \begin{bmatrix} V_{1,1} \\ V_{2,1} \\ \cdots \\ V_{r,1} \\ \mu_1 \\ \cdots \\ V_{1,P} \\ \cdots \\ V_{r,P} \\ \mu_P \end{bmatrix} \tag{EQ 10}$$

Note that H and K are block diagonal matrices.

Because $\mathbf{f}_1$ and $\mathbf{f}_2$ contain the same observables as $\mathbf{f}$,

$$\phi = \frac{\mathbf{f}^T\mathbf{f}}{2} = \frac{\mathbf{f}_1^T\mathbf{f}_1}{2} = \frac{\mathbf{f}_2^T\mathbf{f}_2}{2} \tag{EQ 11}$$

and

$$\dot{\phi} = \begin{bmatrix} H^T H\mathbf{u} - H^T(\mathbf{w}_1 - \hat{\mu}) \\ K^T K\tilde{\mathbf{v}} - K^T\mathbf{w}_2 \end{bmatrix} \qquad \text{(EQ 12)}$$

since

$$H = -\frac{\partial \mathbf{f}_1}{\partial \mathbf{u}},\ K = -\frac{\partial \mathbf{f}_2}{\partial \tilde{\mathbf{v}}}. \qquad \text{(EQ 13)}$$

If data is complete, K is a block diagonal matrix of dimension $FP \times (r+1)P$, whose block elements are U matrices of dimension $F \times (r+1)$, replicated along the diagonal, i.e.,

$$K = \begin{bmatrix} U & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & U \end{bmatrix} \qquad \text{(EQ 14)}$$

$$U = \begin{bmatrix} U_{1,1} & & U_{1,r} & 1 \\ & \ddots & & \\ U_{F,1} & & U_{F,r} & 1 \end{bmatrix} \qquad \text{(EQ 15)}$$

when data is incomplete, the elements associated with the missing data are taken out, resulting in a matrix of dimension $m \times (r+1)P$

$$K = \begin{bmatrix} U_{m_1 \times (r+1)} & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & U_{m_P \times (r+1)} \end{bmatrix} = \begin{bmatrix} U_1 & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & U_p \end{bmatrix} \qquad \text{(EQ 16)}$$

where

$$\sum_{p=1}^{P} m_p = m, \text{ and } m_p = \sum_{f=1}^{F} \gamma_{f,p},$$

$\gamma_{f,p} = 1$ when $W_{f,p}$ is observed otherwise $\gamma_{f,p} = 0$.

Similarly, when data is incomplete, we have the following matrix of dimension $m \times rF$

$$H = \begin{bmatrix} V_{n_1 \times r} & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & V_{n_F \times r} \end{bmatrix} = \begin{bmatrix} V_1 & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & V_F \end{bmatrix} \qquad \text{(EQ 17)}$$

where

$$\sum_{f=1}^{F} n_f = m, \text{ and } n_f = \sum_{p=1}^{P} \gamma_{f,p}.$$

The pseudo-inverse matrices of H and K can be easily computed because of their block diagonal structure,

$$K^{+} = \begin{bmatrix} U_1^{+} & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & U_p^{+} \end{bmatrix} \quad \text{(EQ 18)}$$

$$H^{+} = \begin{bmatrix} V_1^{+} & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & V_F^{+} \end{bmatrix} \quad \text{(EQ 19)}$$

2.4 WLS formulation

The minimization problem (EQ 1) can be generalized as a WLS problem

$$min \quad \phi = \frac{1}{2}\sum_{f,p} (\gamma_{f,p}(W_{f,p} - \mu_p - \mathbf{u}_{f.}^T \mathbf{v}_{p.}))^2 \quad \text{(EQ 20)}$$

where $\gamma_{f,p}$ is weighting factor for each measurement $W_{f,p}$.

In the previous discussion, we have assumed that all weights are either one when data is observable or zero when unobservable. However, in many cases we may prefer to assign different weights other than ones or zeros to individual measurement. For example, in recovering the pose of a 3D plane, we can assign the confidence measurement to each recovered surface normal by its incidence angle with the viewing direction. Different sensor models can be applied to obtain a weighting matrix if necessary. In the following, we formulate principal component analysis with missing data as a WLS problem.

We introduce two $FP \times FP$ diagonal weight matrices,

$$\Gamma = diag(\Gamma_1, \Gamma_2, \ldots, \Gamma_P) \quad \text{(EQ 21)}$$

and

$$\Phi = diag(\Phi_1, \Phi_2, \ldots, \Phi_F) \tag{EQ 22}$$

where

$$\Gamma_p = diag(\gamma_{p,1}, \gamma_{p,2}, \ldots, \gamma_{p,F}),\ p = 1, \ldots, P,$$

and

$$\Phi_f = diag(\gamma_{1,f}, \gamma_{2,f}, \ldots, \gamma_{P,f}),\ f = 1, \ldots, F.$$

The minimization problem becomes

$$min \quad \phi = \frac{\mathbf{f}_{\gamma 1}^T\ \mathbf{f}_{\gamma 1}}{2} = \frac{\mathbf{f}_{\gamma 2}^T\ \mathbf{f}_{\gamma 2}}{2} \tag{EQ 23}$$

where

$$\mathbf{f}_{\gamma 1} = \Gamma\ \mathbf{f}_1 \text{ and } \mathbf{f}_{\gamma 2} = \Phi\ \mathbf{f}_2.$$

The solution to the above problem is when the first order derivative of the objective function becomes zero. The derivative of the objective function is

$$\dot{\phi} = \begin{bmatrix} H_\gamma^T H_\gamma \mathbf{u} - H_\gamma^T(\mathbf{w}_1 - \hat{\mu}) \\ K_\gamma^T K_\gamma \tilde{\mathbf{v}} - K_\gamma^T \mathbf{w}_2 \end{bmatrix} \tag{EQ 24}$$

where

$$H_\gamma = \Gamma H = \begin{bmatrix} \Gamma_1 V & & \\ & \ddots & \\ & & \Gamma_P V \end{bmatrix} \tag{EQ 25}$$

and

$$K_\gamma = \Phi K = \begin{bmatrix} \Phi_1 U & & \\ & \ddots & \\ & & \Phi_F U \end{bmatrix} \tag{EQ 26}$$

Therefore, after computing the pseudo inverses of H_γ and K_γ

$$H_\gamma^+ = \begin{bmatrix} H_{\gamma 1}^+ & & \\ & \dots & \\ & & H_{\gamma P}^+ \end{bmatrix} = \begin{bmatrix} (V^T \Gamma_1^T \Gamma_1 V)^{-1} (V^T \Gamma_1^T) & & \\ & \ddots & \\ & & (V^T \Gamma_P^T \Gamma_P V)^{-1} (V^T \Gamma_P^T) \end{bmatrix} \quad \text{(EQ 27)}$$

$$K_\gamma^+ = \begin{bmatrix} K_{\gamma 1}^+ & & \\ & \dots & \\ & & K_{\gamma F}^+ \end{bmatrix} = \begin{bmatrix} (U^T \Phi_1^T \Phi_1 U)^{-1} (U^T \Phi_1^T) & & \\ & \ddots & \\ & & (U^T \Phi_F^T \Phi_F U)^{-1} (U^T \Phi_F^T) \end{bmatrix} \quad \text{(EQ 28)}$$

we can then use the PCAMD (principal component analysis with missing data) algorithm to solve the WLS problem. Our formulation is essentially a modified NIPALS Ruhe-Wiberg algorithm. The algorithm is as follows:

Algorithm PCAMD

(1) initialize $\tilde{\mathbf{v}}$

(2) update

$$\mathbf{u} = H_\gamma^+ (\mathbf{w_1} - \hat{\mu})$$

(3) update

$$\tilde{\mathbf{v}} = K_{\mathbf{r}}^+ \mathbf{w_2}$$

(4) stop if the algorithm converges, or go back to (2).

Remarks:

(1) Ruhe [15] also suggested using Newton and Gauss methods to speed up the convergence of the NIPALS method. In practice, we found that the NIPALS method converges within desired tolerance in several iterations in our experiments.

(2) Ruhe [15] and Wiberg [22] also showed that the more the missing data, the worse the result will be. It is hardly surprising because the method is basically an interpolation of all observable elements. Statistically this corresponds to decreasing robustness of the estimate for the principal components given the observations. Fortunately object

modeling from multiple views, we can always take many views to form a well constrained problem for our modeling purpose. Determining a minimally acceptable number of views can be regarded as a sensor planning problem.

(3) The missing data can also be extrapolated as long as we find some sub-blocks in the measurement matrix which satisfy *Lemma 1*. The issue of obtaining those blocks is non-trivial. Once the missing data have been augmented, a linear or nonlinear optimization method can be applied to solve the original problem. It should work well if data is noise-free, i.e., only the first r singular values of the reconstructed measurement matrix are non-zero. However, this method becomes of questionable utility when any result from sub-block computation is inaccurate.

(4) There are statistical ways to improve the solution, for example, the metrically Winsorised residuals method [18]. This method is based on the assumption that each measurement is corrupted by additive Gaussian noise. The metrically Winsorised residuals method adjusts the weight for each measurement depending on its residual error.

3 Merging Multiple Views

Principal component analysis with missing data has been formulated to a WLS minimization problem in the previous section and a PCAMD algorithm is proposed to solve it. From the motivational example of a dodecahedron it is clear that object modeling from a sequence of views should be formulated as a WLS problem.

In this section, we show that multiple view merging can be formulated as a combination of two WLS problems. The first WLS problem involves rotation matrices and surface normals, which is independent of translation. Once the first problem is solved, the second WLS problem yields translation vectors and normal distances to the origin. The first WLS problem of determining rotation matrices and surface normals can be further simplified by representing the rotation matrix using the quaternion. A straightforward two-step iterative algorithm can be devised to solve these two problems using the PCAMD algorithm from the previous section.

3.1 Two WLS problems

Suppose that we have tracked P planar regions over F frames. We then have trajectories of plane coordinates $\{(\mathbf{v}_{fp}, d_{fp}) \mid f = 1, \ldots, F, p = 1, \ldots, P\}$ where $\mathbf{v}_{fp}$ is the surface normal of the pth patch in the fth frame, and d_{fp} is the associated normal distance to the origin. To facilitate the decomposability of rotation and translation, instead of forming a $4F \times P$ measurement matrix as in section 2.1, we form surface normals v_{fp} into a $3F \times P$ matrix $W^{(v)}$ and distances d_{fp} into an $F \times P$ matrix $W^{(d)}$. $W^{(v)}$ and $W^{(d)}$ are called the *normal measurement matrix* and *distance measurement matrix* respectively.

It can be easily shown that $W^{(v)}$ has at most rank 3 and $W^{(d)}$ has at most rank 4 when noise-free, therefore, $W^{(v)}$ and $W^{(d)}$ are highly rank-deficient. We decompose $W^{(v)}$ into

$$W^{(v)} = R\ V \qquad \text{(EQ 29)}$$

where

$$R = \begin{bmatrix} R^{(1)} \\ \ldots \\ R^{(F)} \end{bmatrix}$$

is the rotation matrix of each view with respect to the world coordinate system, and $V = [\mathbf{v}_1, \ldots, \mathbf{v}_P]$ is the surface normal matrix in the world coordinate system. Since R is a $3F \times 3$ matrix and V is an $3 \times P$ matrix, the rank of $W^{(v)}$ is at most 3.

Similarly, we can decompose $W^{(d)}$ into

$$W^{(d)} = T\ M \qquad \text{(EQ 30)}$$

where

$$M = \begin{bmatrix} \begin{bmatrix} \mathbf{v}_1 \\ d_1 \end{bmatrix} & \ldots & \begin{bmatrix} \mathbf{v}_P \\ d_P \end{bmatrix} \end{bmatrix}, \text{ and } T = \begin{bmatrix} \begin{bmatrix} \mathbf{t}_1 R_1 & 1 \end{bmatrix} \\ \ldots \\ \begin{bmatrix} \mathbf{t}_F R_F & 1 \end{bmatrix} \end{bmatrix}$$

$\mathbf{t}_f$ and R_f are the translation vector and rotation matrix of view f with respect to a fixed world coordinate system.

Note that decomposition of $W^{(d)}$ depends on the decomposition of $W^{(v)}$. Since M is $4 \times P$ and T is $F \times 4$, the rank of $W^{(d)}$ is at most 4.

We can also decompose $W^{(d)}$ into

$$W^{(d)} = \begin{bmatrix} t_1 R_1 \\ \ldots \\ t_F R_F \end{bmatrix} \begin{bmatrix} v_1 & \ldots & v_P \end{bmatrix} + \begin{bmatrix} 1 \\ \ldots \\ 1 \end{bmatrix} \begin{bmatrix} d_1 & \ldots & d_P \end{bmatrix} \qquad \text{(EQ 31)}$$

When all elements in the two measurement matrices are known, we need to solve two least-squares problems. However, since only part of the planar regions are visible in each view, we end up with two WLS problems instead. The first least squares problem, labeled as WLS-1, is

$$min \sum_{f=1,\ldots,F,p=1,\ldots,P} (\gamma^{(v)}_{f,p} (W^{(v)}_{f,p} - [RV]_{f,p}))^2 \qquad \text{(EQ 32)}$$

and the second one, denoted as WLS-2, is

$$min \sum_{f=1,\ldots,F,p=1,\ldots,P} (\gamma^{(d)}_{f,p} (W^{(d)}_{f,p} - [TM]_{f,p}))^2 \qquad \text{(EQ 33)}$$

where

$\gamma_{f,p} = 0$ if surface p is invisible in frame f, and $\gamma_{f,p} = 1$ otherwise. All weights can be any number between zero and one, depending on the significance or confidence of each measurement. A similar WLS formulation is also used in [18].

3.2 Quaternion WLS-1

It appears, from the last section, that we can devise a naive two-step algorithm which solves WLS-1 and subsequently WLS-2 by applying the PCAMD algorithm to both problems. However, in order to solve WLS-1, we iterate $R^{(f)}$ as if it had nine independent parameters, while it is a nonlinear trigonometric function of three parameters. Although it is possible to normalize $R^{(f)}$ after every iteration, it may perform poorly in terms of robustness and efficiency.

In fact, several representations of rotation are often used in practice:

(1) An orthonormal rotation matrix R

(2) An rotation axis **a** and an rotation angle θ

(3) A unit quaternion q

A quaternion is a 4-tuple (**w**,s) where **w** is a 3-vector and s is a scalar. The mapping between a unit quaternion and a rotation axis along with a rotation angle is given by $\mathbf{w} = \sin(\theta/2)\,\mathbf{a}$ and $s = \cos(\theta/2)$. The quaternion representation of the rotation matrix leads to a simple way of solving minimization problems of 3D point matching and surface normal matching, as demonstrated in [8].

The WLS-1 problem (EQ 32) can be decomposed into F minimization problems

$$min \quad \sum_{p=1}^{P} \gamma_f^{(v)} \left\| \mathbf{w}_f - R^{(f)} \mathbf{v}_p \right\| \qquad \text{(EQ 34)}$$

where

$$f = 1, \ldots, F,\ \mathbf{w}_f = [W_{f,1}, \ldots, W_{f,P}]^T,\ \gamma_f^{(v)} = diag(\gamma_{f,1}, \ldots, \gamma_{f,P}).$$

The above problem can be reformulated using quaternion as

$$\min \quad \sum_{p=1}^{P} \gamma_f^{(v)} \left\| \mathbf{w}_f - q^{(f)} \mathbf{v}_p \tilde{q}^{(f)} \right\| \qquad \text{(EQ 35)}$$

$$\textit{subject to} \qquad |q| = 1$$

where $\tilde{q}$ is the conjugate quaternion of q, and

$$\sum_{p=1}^{P} \gamma_f^{(v)} \left\| \mathbf{w}_f - q^{(f)} \mathbf{v}_p \tilde{q}^{(f)} \right\| = \sum_{p=1}^{P} \gamma_f^{(v)} \left\| \mathbf{w}_f q^{(f)} - q^{(f)} \mathbf{v}_p \right\| = \sum_{p=1}^{P} q^{(f)^T} A_p^{(f)} q^{(f)} \qquad \text{(EQ 36)}$$

$A_p^{(f)}$ are symmetric matrices because $\mathbf{w}_f q^{(f)} - q^{(f)} \mathbf{v}_p$ is a linear function of $q^{(f)}$. Obviously

$$B^{(f)} = \sum_{p=1}^{P} A_p^{(f)} \qquad \text{(EQ 37)}$$

is also symmetric, and the minimization problem () becomes

$$\min \quad q^{(f)^T} B^{(f)} q^{(f)} \qquad \text{(EQ 38)}$$

The solution to the above minimization problem is the eigenvector $q_{min}^{(f)}$ corresponding to the minimum eigenvalue of the matrix $B^{(f)}$.

3.3 Iterative algorithm

We combine quaternion-based rotation matrix updating to form a two-step algorithm to solve both the first and the second WLS problems. The algorithm is as follows:

Algorithm two-step WLS's

Step 0 Initialization

(0.1) read in measurement matrices $\mathbf{W}^{(\mathbf{v})}$, $\mathbf{W}^{(\mathbf{d})}$

(0.2) read in weight matrices $\gamma^{(\mathbf{v})}$, $\gamma^{(\mathbf{d})}$

(0.3) initialize R, vectorize R to v

Step 1 WLS-1

(1.1) vectorize $\mathbf{W}^{(\mathbf{v})}$ **to** $\mathbf{w}_{\mathbf{v1}}$ **and** $\mathbf{w}_{\mathbf{v2}}$

(1.2) update $\mathbf{H}^{+}_{\mathbf{v}\gamma}$

(1.3) update

$$\mathbf{u} = \mathbf{H}^{+}_{\mathbf{v}\gamma}\mathbf{w}_{\mathbf{v1}}$$

(1.4) update

$$\mathbf{B}^{(\mathbf{f})}$$

(1.5) update

$$\mathbf{q}^{(\mathbf{f})}$$

and transform to R, vectorize to v

(1.6) go to (1.2) if not converged, otherwise advance to Step 2

Step 2 WLS-2

(2.1) vectorize $\mathbf{W}^{(\mathbf{d})}$ **to** $\mathbf{w}_{\mathbf{d1}}$ **and** $\mathbf{w}_{\mathbf{d2}}$

(2.2) update $\mathbf{H}^{+}_{\mathbf{d}\gamma}$

(2.3) update

$$\mathbf{u} = \mathbf{H}^{+}_{\mathbf{d}\gamma}\mathbf{w}_{\mathbf{d1}}$$

(2.4) update

$$\mathbf{K}^{+}_{\mathbf{d}\gamma}$$

(2.5) update

$$\tilde{\mathbf{v}} = \mathbf{K}^{+}_{\mathbf{d}\gamma}\mathbf{w}_{\mathbf{d2}}$$

(2.6.) stop if converged, otherwise go to (2.2).

We have not explicitly discussed the normalization problem in our WLS approach. The normalization problem occurs because the measurement matrix is rank-deficient, hence, there are infinite solutions to the minimization problem (EQ 1) unless an additional constraint is imposed. This additional constraint is generally problem dependent; for example, the 2-norm of the factorized left matrix is constrained to be one [15]. Fortunately, we have implicitly constrained our rotation matrices in its quaternion representation. The remaining constraint in the first WLS is the normalization of surface normal vectors which are constrained to be of unit magnitudes.

Prior to multiple view merging, we need to track surfaces so that normal measurement matrix and distance measurement matrix can be formed. The next section describes our surface tracking algorithm.

4 Surface Patch Tracking

In this section, we briefly overview each module of our surface patch tracking system: range image segmentation, adjacency graph building, and two-view matching.

4.1 Range image segmentation

There are many different techniques for range image segmentation. By and large they can be divided into feature-based and primitive-based approaches, although statistics-based approaches have also been introduced recently. Feature-based approaches yield precise segmentation but are sensitive to noise in practice. For example, Gaussian and mean curvatures can be used to label different regions before region growing; however, this process is quite sensitive to noise because of the second order derivative. Primitive-based approaches are more robust to noise but constrained by the number of primitives. The higher the degree of surface polynomial, the more difficult and the less robust the segmentation is likely to be.

We have used the primitive-based region growing segmentation method of [8]. The two types of surface primitives used are the planar surface and the quadric surface. The regions are established via region growing from seed points, i.e., the seed points are chosen from points which are closest to their approximating primitives, and then merged with their neighbors until the best-fit errors become unacceptable.

4.2 Adjacency graph

Once we have successfully segmented the range data for each view, the range image associated with view i can be represented as a set of planar regions $I_i = \{v_{ij}, d_{ij}, c_{ij}\}$, where v_{ij} and d_{ij} are the normal and distance of the jth segment planar surface respectively, and c_{ij} is the centroid of jth segmented region.

From each view of the 3D object, we build an adjacency graph where every node in the graph represents a visible planar region and each arc connects two adjacent nodes. The adjacency graph is updated whenever this view is matched with another. Eventually we have adjacency information among all visible planar regions after tracking all of them for the whole sequence. From the adjacency graph, all the object vertices can be located; thus, 3D object model is obtained. However, augmenting adjacency graph is difficult for concave objects because of occlusion. A better way of establishing spatial connectivity among all surfaces is discussed in section 5.

We have implemented the planar surface patch tracking system which employs a wavefront algorithm to generate the adjacency graph. The wavefront algorithm makes use of range data because there is significant change in range data across an occluding edge.

4.3 Matching two views

Given two adjacent segmented images I_1 and I_2, we would like to find correspondence between different regions in two views, i.e., we want to find a mapping $\phi: \quad (I_1 \rightarrow I_2)$ such that a certain distance measurement $d(I_1, I_2)$ is minimized.

Two questions arise in matching two views of planar regions. The first is how to make correspondence between two views; the second is how to recover the transformation between them. Our solution to the first problem is to use adjacency information between two segmented patches and between segmented surface normals. If displacement between two views is relatively small, there should be only linear shape change [11] within the same aspect, corresponding segmented regions are of similar size (number of points), centroid, and surface normals. When a new aspect appears, which signals a nonlinear shape change, there would be significant change in these parameters. There may not always be solutions to the second problem because we need at least two corresponding non-parallel faces to determine rotation and three to determine translation. In practice, we can make the assumption that we always have two non-parallel corresponding faces in two adjacent views.

In fact, solving the second problem can be of help to the first problem because we can then make use of the hypothesis-and-test approach. We iteratively select two pairs of non-parallel faces from the two images to be matched, estimate the corresponding rotation matrix, and then attempt to match the rest of the faces. We always choose two adjacent faces from both images, and match them based on surface normal, distance and centroid of segment regions. The number of faces matched and consistency in face adjacency are used in the distance measure between two matches. The estimated transformation matrix is only used to help building the adjacency graph, while the precise transformation is robustly recovered from our WLS method.

Multiple view tracking is done by sequentially matching two adjacent views. Whenever a new view is added, the adjacency graph and the weight matrix are automatically modified. Because of the problems associated with updating adjacency graph, subsequent to surface patch tracking and multiple view merging, we use another algorithm to establish the spatial connectivity among surfaces.

5 Spatial Connectivity

Once we have extracted the equations of planar surfaces of the object, we then need to establish spatial connectivity relationship among these surfaces. One approach is to build adjacency graph from a sequence of views, as discussed in previous section. However, augmenting adjacency graph whenever a new view is introduced, is quite ad-hoc. In this section, we present a new approach of recovering surface connectivity after all surface patches are recovered. We show that the problem of spatial connectivity of boundary surfaces can be reduced to one of connectivity of supporting lines of a simple polygon.

5.1 Half-space intersection and union

We assume that every planar patch *P* of object model is a simple polygon. A simple polygon does not self-intersect. Every (infinite) plane divides the space into two parts, inside and outside, with surface normal pointing towards the external side of the object. Given an unbounded planar surface, if we intersect all other planar surfaces on it, we obtain supporting lines as illustrated in Fig.2. Each supporting line is directed so that the interior of *P* lies locally to its right. The right half-plane created by such a directed supporting line *c* is called the supporting half-plane, and is characterized as supporting the polygon [6]; however, a concave *P* might not all lie in the right half-plane as indicated in Fig.2.

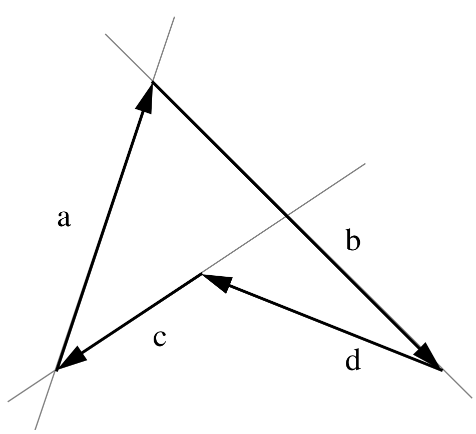


Figure 2 **A simple polygon and its supporting lines (stippled and solid lines)**

For each point x in the plane, if we know which side of each supporting line x lies on, we know if x is inside P. Therefore, the polygon P (and its interior) can be represented as a boolean formula whose atoms are those supporting lines. In other words, a simple polygon can be represented by intersection and union of its supporting line. For example, a boolean formula for the polygon in Fig.2 can be $\overline{c\bar{a} \oplus a\bar{b} \oplus b\bar{d} \oplus \bar{d}c}$. This Guibas style [6] formula is obtained by complementing the second supporting line at a convex angle, and the first supporting line at a concave angle when we go around the polygon. Other boolean formulas such as Peterson style are also possible [6]. Once spatial connectivity is established, the Guibas style formula is straightforward.

5.2 Modified Jarvis' march

The problem of establishing spatial connectivity of supporting lines can be formulated as a modified convex hull-like problem which involves only vertices. This problem can also be regarded as one of cell decomposition which involves data points. We propose a modified Jarvis' march algorithm to reconstruct simple polygons from supporting lines and valid data points. The algorithm to recover spatial connectivity among 3D surfaces is discussed in section 5.3.

Definition 1

A point is defined as valid in a simple polygon if there exist sufficient valid data points around its neighborhood.

Lemma 2

The intersection point P of two supporting lines is valid in a simple polygon if and only if the intersection of two corresponding half-planes is valid locally at P.

Proof:

When the intersection of two half-planes is valid locally at P, the intersection point of these two supporting lines is valid by its definition.

Assume that the intersection point of two supporting lines is valid. Since two lines divide the plane into four regions, there must exist at least one such region among four around the intersection point that is a valid cell of the simple polygon. Therefore, the intersection of two half-planes is valid locally at *P*. (Q.E.D.)

The *Lemma 2* leads to a modified Jarvis' march algorithm of reconstructing simple polygon from supporting lines and valid data points.

To construct the simple polygon from all supporting lines and valid data points, we first pre-compute all intersection points which are candidates of vertices of the simple polygon. If we march successive vertices with the least turning angle, we obtain their convex hull; this is referred to as Jarvis' march algorithm [14]. The kernel of the simple polygon, if it exists, can also be found by intersecting all half-spaces. Using *Lemma 2*, however, enables us to find the correct simple polygon by marching all points whose local neighborhood is valid. We call this algorithm the "modified Jarvis' march".

Assume that we have first found the lowest left point *p1* of the set of vertex candidates, which is certainly a convex hull vertex, but not necessarily a vertex for our simple polygon (unless it is valid locally). For example, in Fig.3, *p1* is not a simple vertex because *p5p1p2* is not a valid triangle cell (valid cells are shaded areas which show valid data points). Since *p6p2p3* is a valid triangle cell, we start our algorithm from *p2*.

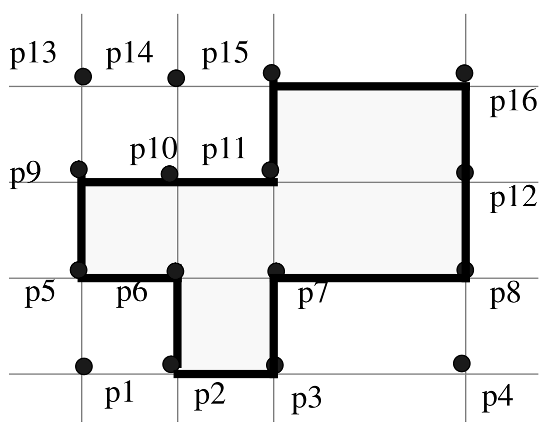


Figure 3 **Example of modified Jarvis' march and cell decomposition. Shaded area represents valid data points.**

A data structure is defined for each intersection point *P* as follows

typedef {
  intersect-point left, right, up, down;
  intersect-point previous, next;
} **intersect-point** **P**;

Fig.4 shows the relationship among the members of the data structure. Assume that an intersection point is intersected by only two supporting lines.

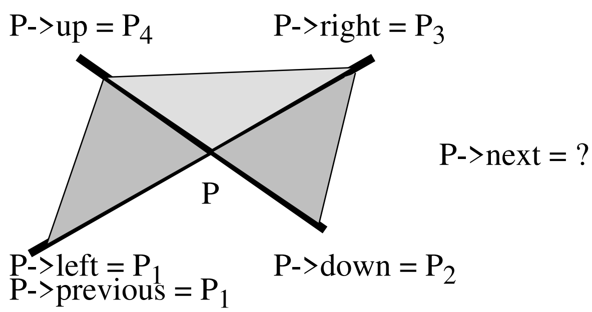


Figure 4 **Illustration of data structure of intersection point**

After the starting vertex is found, we march for the next vertex as illustrated in Fig.4. If there are sufficient data points in cell *PP2P3*, next valid vertex is *P2*; If *P2* is not valid, we check if *P3* is valid; If *P3* is also invalid, *P4* must be valid, or an error will occur. The march ends when the next vertex is the starting vertex. The modified Jarvis' march (MJM) algorithm is given as follows:

Algorithm MJM

Step 1. initialize starting vertex

START->previous = **NULL**,
P = **START**->next,
P->previous = **START**;

Step 2. march

P->left = $\mathbf{P_1}$; **P**->down = $\mathbf{P_2}$; **P**->right = $\mathbf{P_3}$; **P**->up = $\mathbf{P_4}$;
if cell $\mathbf{PP_2P_3}$ **valid,** **P**->next = $\mathbf{P_2}$ (case 1)
else if cell $\mathbf{PP_3P_4}$ **valid,** **P**->next = $\mathbf{P_3}$ (case 2)
else if cell $\mathbf{PP_4P_1}$ **valid,** **P**->next = $\mathbf{P_4}$ (case 3)
else error occurs;

Step 3. terminate

if P->next = **START**.

A postprocessing step may be necessary to remove points which belong to case 2 in step 2 of the march algorithm. These points are on the same line with its previous point and its next point. For example, in Fig.3, *p12* can be removed because *p8* and *p16* make it redundant.

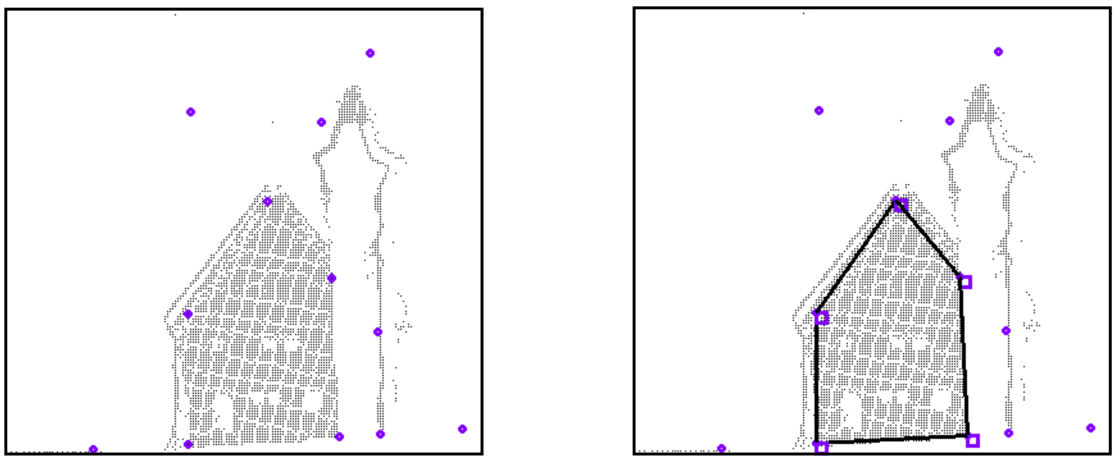
As can be seen from the above algorithm and Fig.3 as well as Fig.4, the problem of single polygon reconstruction from supporting lines and valid data points is one of cell decomposition. As we march around all supporting lines, the Guibas style boolean formula of the simple polygon can be readily formulated.

5.3 3D spatial connectivity

So far we have discussed the problem of recovering the connectivity of supporting lines of a simple polygon. The approach uses information at both signal level (real data points) and algebraic level (line equations). The same hybrid approach can be applied to the problem of spatial connectivity of planar surfaces in 3D.

Indeed, the problem of connectivity of planar surfaces in 3D can be reduced to a set of problems of connectivity in 2D. Assume that we have recovered a set of *N* face equations and transformation among different views (e.g., from *PCAMD*). All valid data points from multiple views can be merged in the same world coordinate system. For each face F_i, if we intersect all other *N-1* faces F_j ($j = 1, \ldots, N-1, j \neq i$). with F_i and project all these lines onto F_i, we get *M (=N-1)* supporting lines on face F_i. We also project nearby 3D points onto this face F_i. Without loss of generality, we assume that no two supporting lines are parallel (or a normal threshold *d* can be set such that $v_i v_j \geq d$). For any of the *M* supporting lines, we intersect it with the rest *M-1* lines, we get all possible candidates for vertices of the valid simple polygon which is the model of face F_i, as illustrated in Fig.5. The modified Jarvis' march algorithm can be then applied to each of the *N* faces accordingly. By connecting all polygons recovered, we get the entire 3D object model boundary. A simple algorithm can be accordingly constructed to establish 3D spatial connectivity then.

Fig. 5 shows an example. It is a face of a toy house model. The complete house model is reconstructed and presented in next section. Fig.5(a) shows intersections of supporting lines and nearby data points projected on this face, while Fig.5(b) superimposes a reconstructed simple polygon model of this face on Fig.5(a).



(a) (b)

Figure 5 **Reconstruction of connectivity. The tiny dots represent projected nearby data points. Intersections of supporting lines are represented by black circles. Vertices of reconstructed simple polygon are represented by small squares.**

6 Experiments

In this section, we present results of applying our algorithm on synthetic data and on real range image sequence of objects. We demonstrate the applicability and the robustness of our approach using synthetic data, and present the recovered model from real range images.

6.1 Synthetic data

Our synthetic data set consists of a set of 12 planes as in the case of the dodecahedron in the last section. A dodecahedron with 4 different views is shown in Fig.1 in section 2.

6.1.1 Applicability

In this section we study the applicability of the proposed approach. In order to recover the shape of a dodecahedron, given correspondence, how many views are necessary?

For example, we pick 4 distinct views from the viewing sphere so that there is no singularity. Singularity occurs when less than 6 faces are visible. For example, we can formulate two measurement matrices for surface normals and planar distances as follows:

$$W^{(v)} = \begin{bmatrix} \mathbf{v}_1^{(1)} & \mathbf{v}_2^{(1)} & \mathbf{v}_3^{(1)} & \mathbf{v}_4^{(1)} & \mathbf{v}_5^{(1)} & \mathbf{v}_6^{(1)} & * & * & * & * & * & * \\ \mathbf{v}_1^{(2)} & \mathbf{v}_2^{(2)} & \mathbf{v}_3^{(2)} & \mathbf{v}_4^{(2)} & * & * & \mathbf{v}_7^{(2)} & \mathbf{v}_8^{(2)} & * & * & * & * \\ \mathbf{v}_1^{(3)} & \mathbf{v}_2^{(3)} & * & * & * & \mathbf{v}_6^{(3)} & * & \mathbf{v}_8^{(3)} & \mathbf{v}_9^{(3)} & \mathbf{v}_{10}^{(3)} & * & * \\ * & * & * & * & * & * & \mathbf{v}_7^{(4)} & \mathbf{v}_8^{(4)} & \mathbf{v}_9^{(4)} & \mathbf{v}_{10}^{(4)} & \mathbf{v}_{11}^{(4)} & \mathbf{v}_{12}^{(4)} \end{bmatrix} \quad \text{(EQ 39)}$$

$$W^{(d)} = \begin{bmatrix} d_1^{(1)} & d_2^{(1)} & d_3^{(1)} & d_4^{(1)} & d_5^{(1)} & d_6^{(1)} & * & * & * & * & * & * \\ d_1^{(2)} & d_2^{(2)} & d_3^{(2)} & d_4^{(2)} & * & * & d_7^{(2)} & d_8^{(2)} & * & * & * & * \\ d_1^{(3)} & d_2^{(3)} & * & * & * & d_6^{(3)} & * & d_8^{(3)} & d_9^{(3)} & d_{10}^{(3)} & * & * \\ * & * & * & * & * & d_6^{(4)} & * & d_8^{(4)} & d_9^{(4)} & d_{10}^{(4)} & d_{11}^{(4)} & d_{12}^{(4)} \end{bmatrix} \quad \text{(EQ 40)}$$

In order to solve the first WLS problem uniquely for F frames, we need

$$18F \geq 3F + 3P$$

since we have $18F$ equations and $3F$ unknowns for rotation matrices and $3P$ unknowns for surface normals. For the second problem, we have $6F$ equations, but there are only F

unknown translation vectors and P unknown plane distances after using results from the first problem. Therefore, the necessary condition to uniquely solve the second problem is

$$6F \geq 3F + P$$

Since P is *12*, $F \geq 4$ is the unique solution for both problems. Again, we are not concerned with the normalization problem here.

6.1.2 Robustness

We study the effectiveness of our approach when data is corrupted by noise and mismatching occurs. Our synthetic data consists of a set of 12 surface patches randomly distributed around all faces of a dodecahedron. Correspondence is assumed to be known. Only the first WLS problem is studied because of the similarity between those two WLS problems. The minimization of weighted squares distance between reconstructed and given measurement matrices leads to the recovery of surface equations and transformations.

To study the error sensitivity on reconstruction of our algorithm, we take four nonsingular views of the dodecahedron where each component in every surface normal is corrupted by a Gaussian noise of zero-mean and variable standard deviation. As we have shown in the previous section, at least 4 views are required to recover the dodecahedron model. Fig 6 shows that our algorithm converges in a few steps. The cases with standard deviation σ of 0.05, 0.1, 0.2, and 0.5 are studied. Notice that the case with standard deviation of 0.5 yields very noisy original data. As we take more views, the sum of weighted squares error is reduced. Fig. 7 plots the normalized weighted least squares error for 4 views, 8 views, 12 views, and 16 views respectively, while gaussian noise with 0.5 standard deviation is present. The squares errors are normalized because the number of observations increases as more views are introduced.

The more interesting case is when mismatching occurs. Obviously if the face appears only once in the whole sequence, then its reconstruction depends on the amount of noise. When this face appears in more and more views, its reconstruction using our WLS method is averaged over these views. Fig. 8 gives the reconstructed errors of a face which appeared 12 times in 16 views. When only two views are matched, the reconstructed surface normal is deviated from its normal by 18.2 and 38.9 degrees when standard deviation σ is 0.1 and 0.2, respectively. When more views are added, the angle between the reconstructed surface normal and its normal decreases to around 10 and 20 degrees respectively.

When an observed surface normal is wrong in one particular view, the conventional sequential reconstruction method results in an erroneous recovered surface normal and transformation. The errors propagate as new views are introduced regardless of the number of views in which this surface is visible. However, our WLS approach gives appreciably smaller reconstruction error on this observed surface normal by distributing the errors in all views. In any case, in general, our approach cannot be worse than the sequential approach. Fig. 9 compares the reconstructed errors of sequential method and WLS. There are 12 observations of this surface normal in 16 views and its first observation is off by an angle between 0^o and 40^o. The reconstructed models of sequential method and WLS method are shown in Fig. 6 along with the original model, for the case of 40^o angle deviation of one surface normal in the first view. Fig. 10a shows a badly-skewed model, which is the worst case from the sequential method since the error was introduced in the first frame. Fig. 10b shows the reconstructed model by the WLS method while the original dodecahedron model is presented in Fig. 10c.

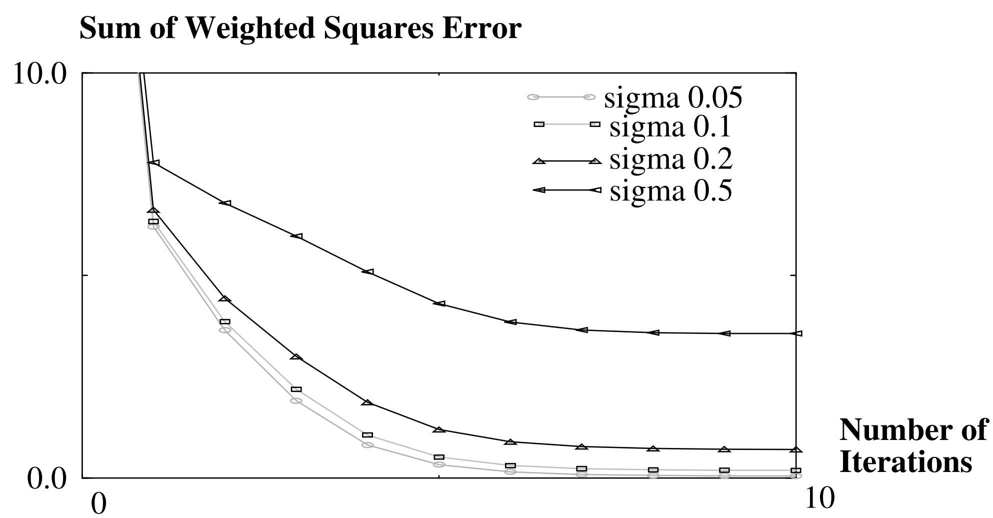


Figure 6 **Effect of noise**

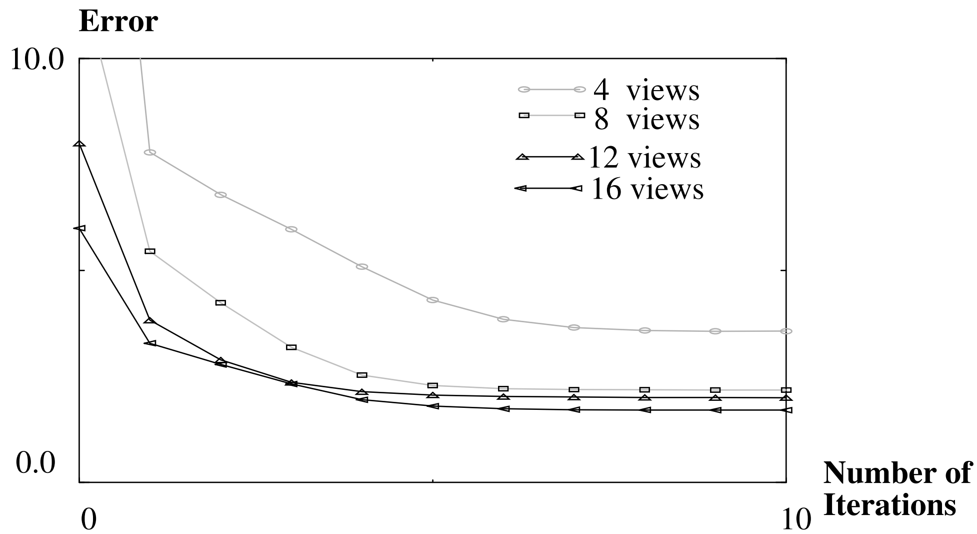


Figure 7 **Effect of number of views**

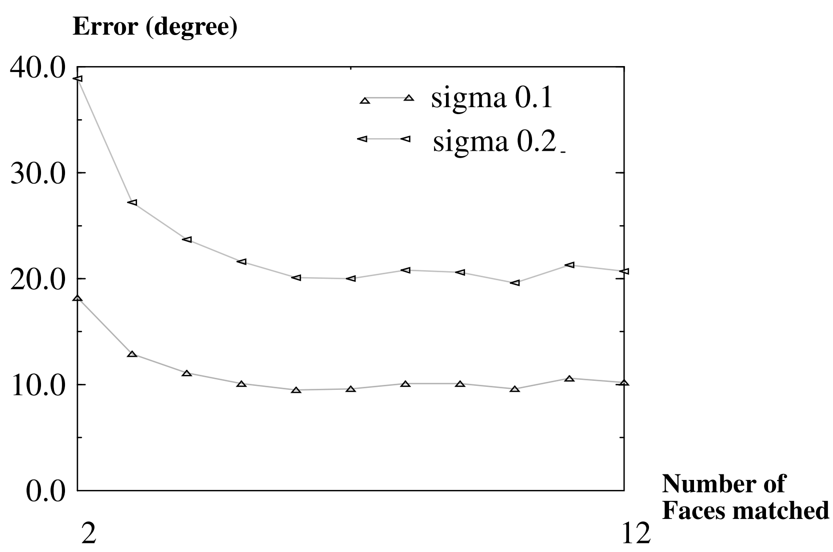


Figure 8 **Reconstructed error vs. number of matched faces**

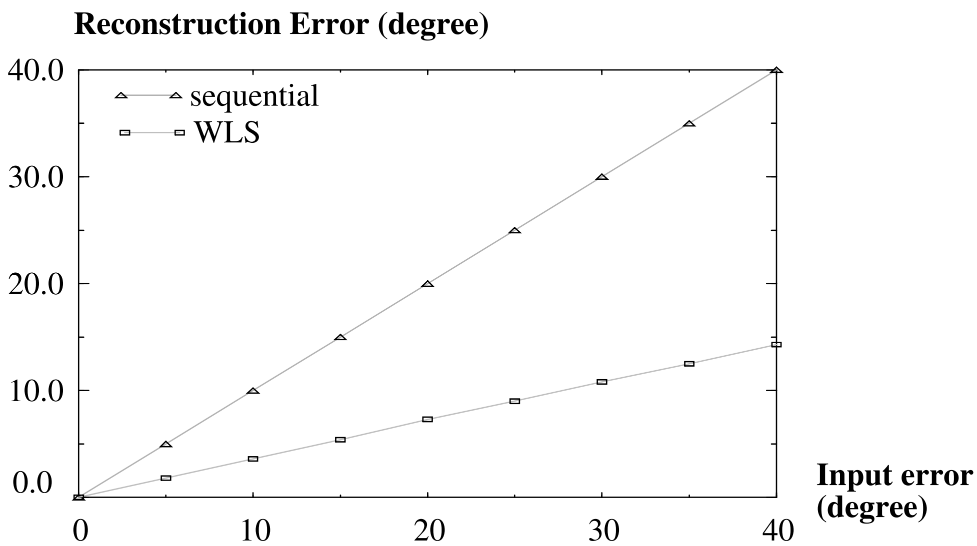


Figure 9 **Comparison between sequential reconstruction and WLS method**

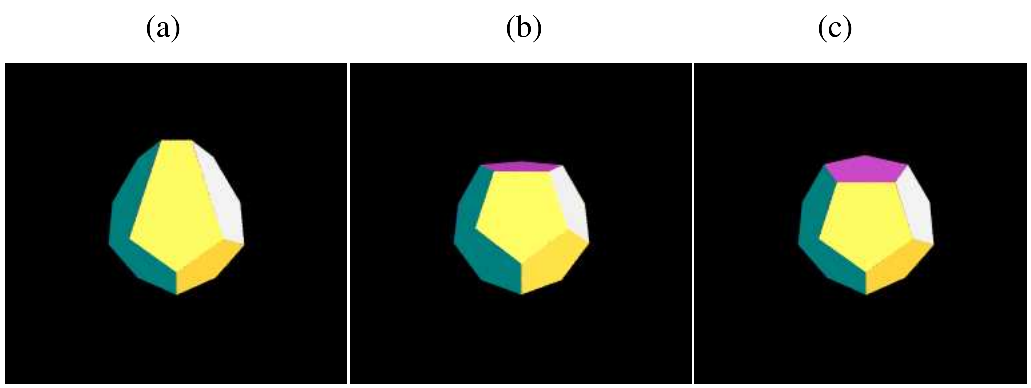
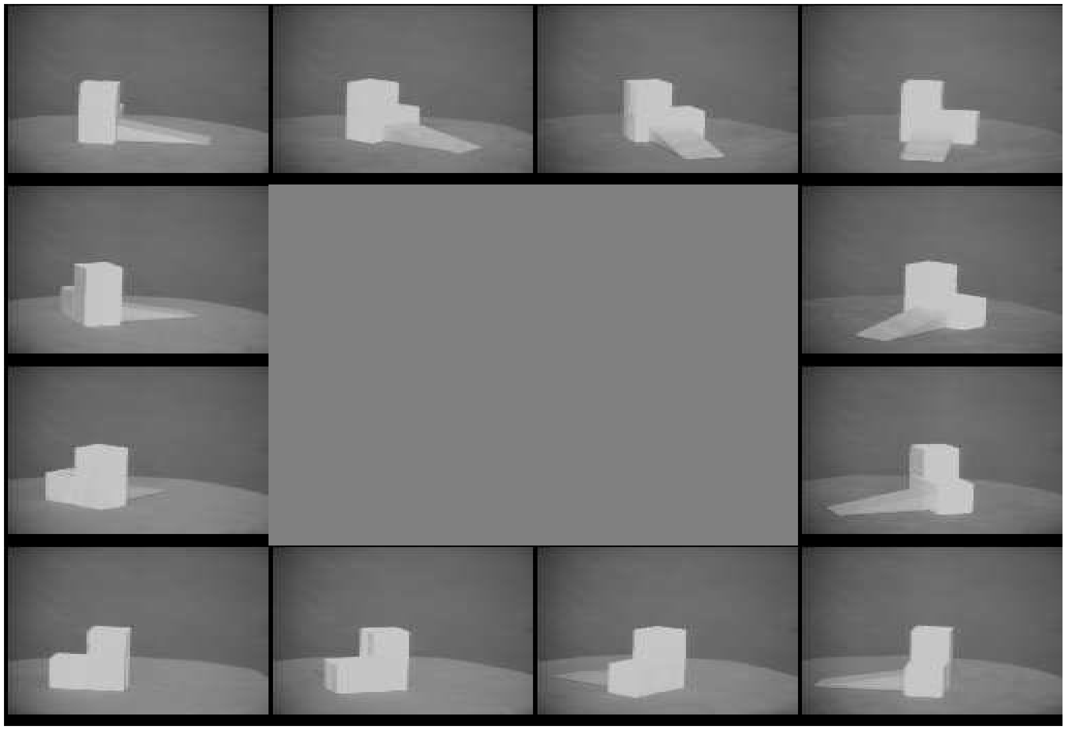


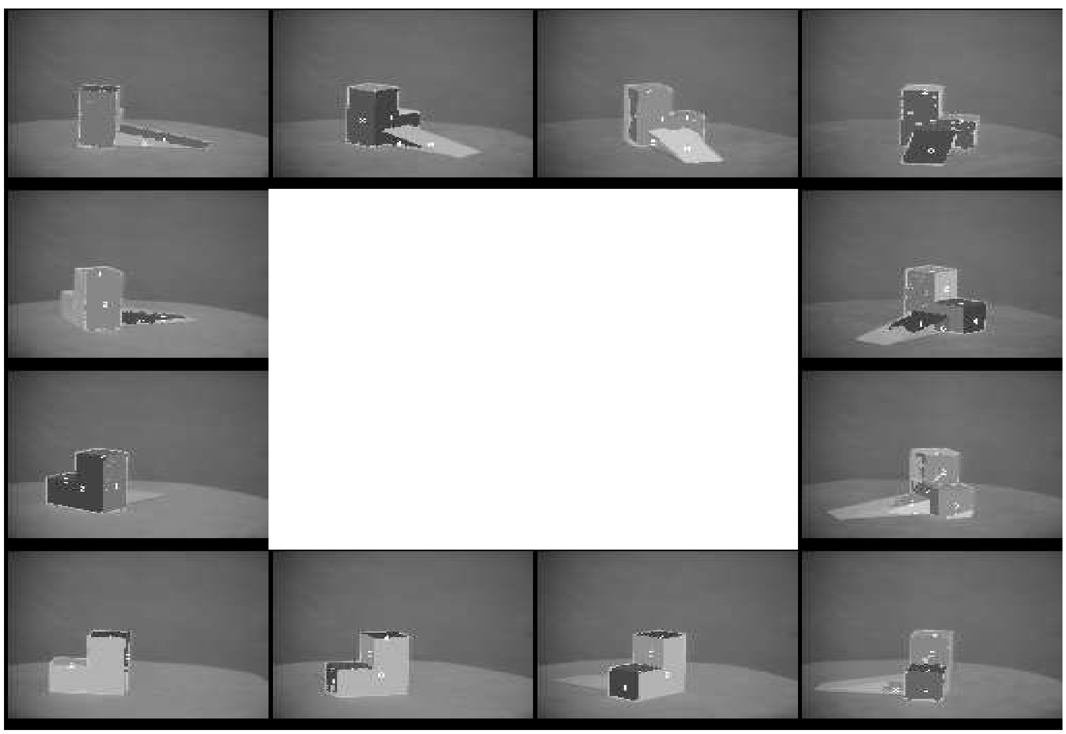
Figure 10 **Recovered and original dodecahedron models (a) worst case of sequential method; (b) our WLS method; (c) original model**

6.2 Real range image sequence

We have applied our algorithm to a sequence of range images of a polyhedral object, using the planar region tracker described in section 3. Figures 11(a) and 11(b) show the whole sequence of 12 views and their corresponding segmentation results. Segmentation is not perfect in several views. Figure 12 shows the result of our system, two shaded views of recovered object model. Figures 13 and 14 show another example of a toy house.



(a)



(b)

Figure 11 **A sequence of images of a polyhedral object (a) original images; (b) after segmentation**

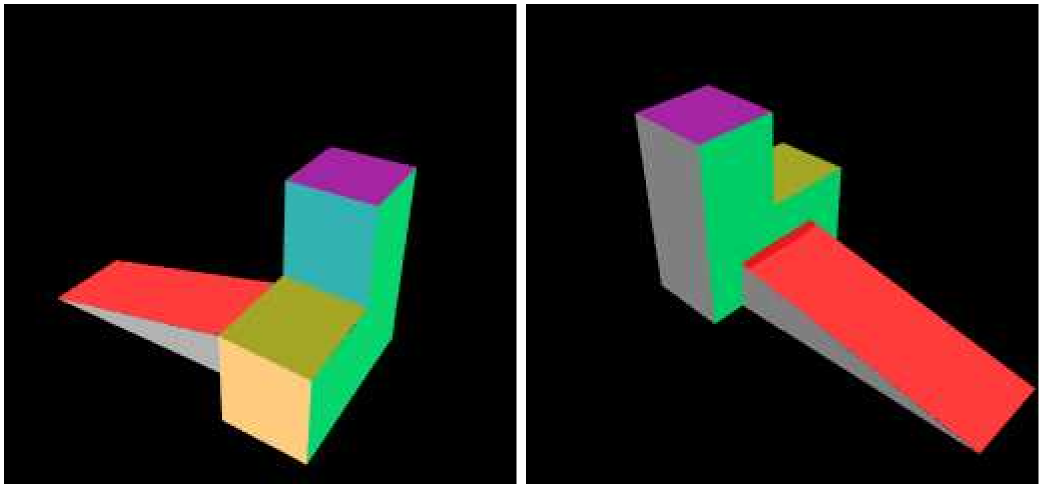


Figure 12 **Two views of shaded display of a recovered model**

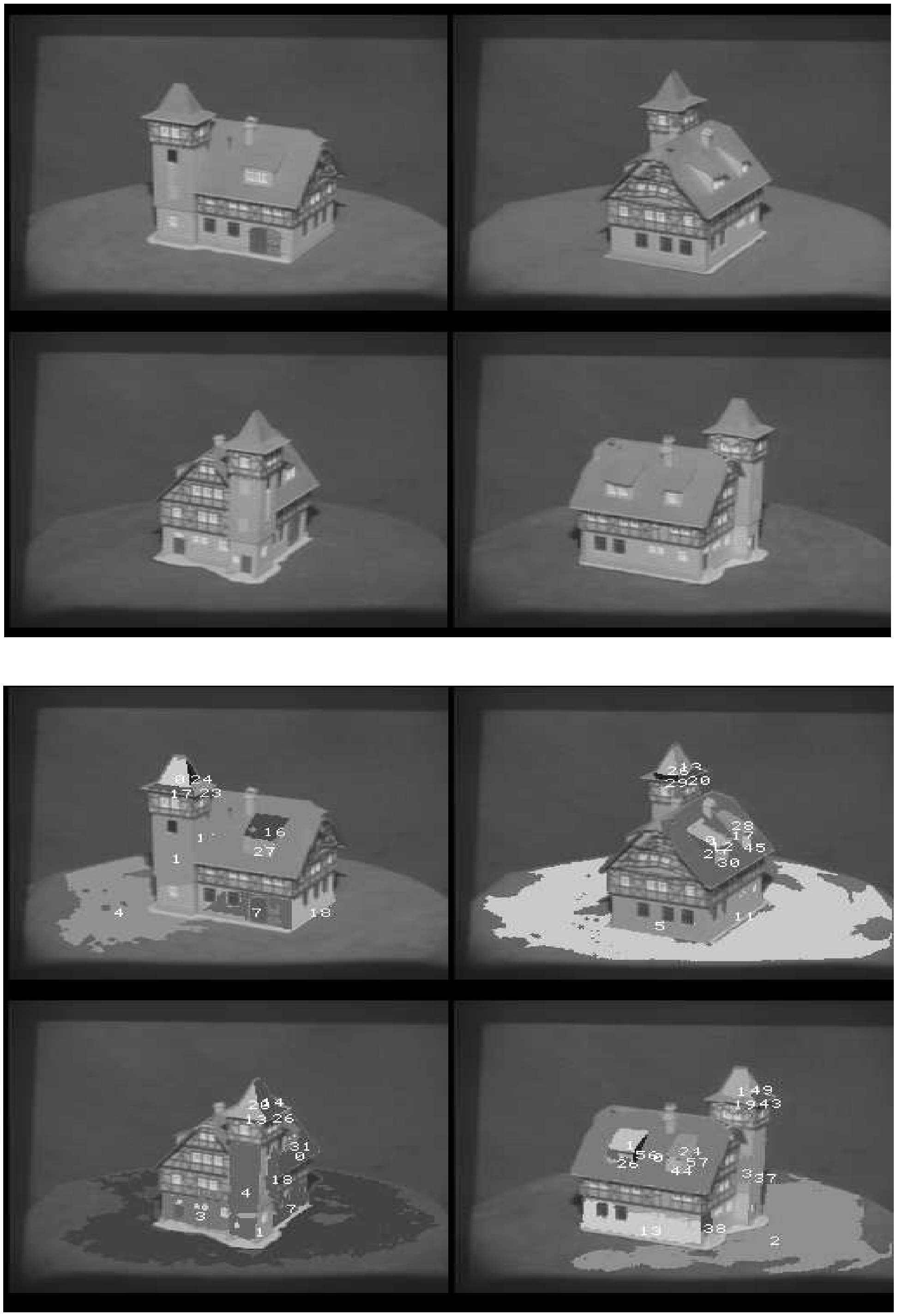


Figure 13 **A sequence of images of a toy house (a) original images; (b) after segmentation**

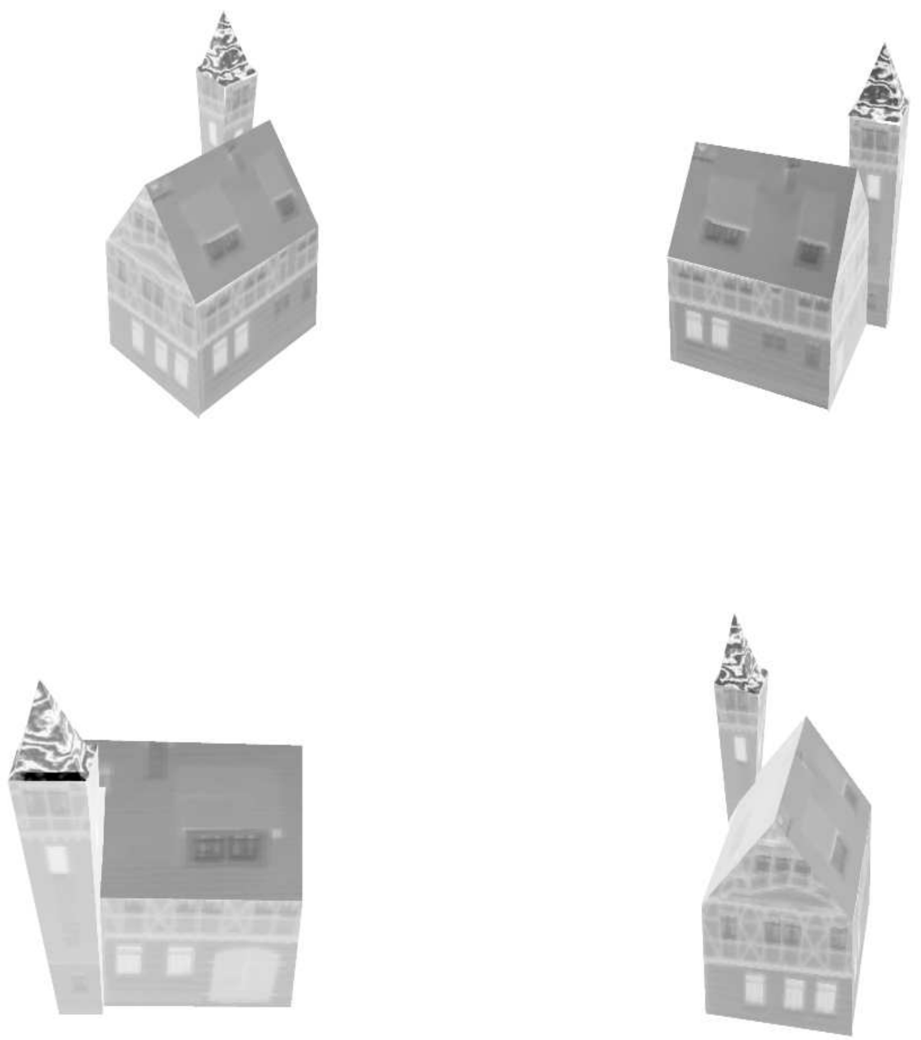


Figure 14 **Four views of texture mapped display of a reconstructed house model**

7 Concluding Remarks

An object modeling approach using multiple range images has been described in this paper. The boundary representation object model is recovered and integrated from different views. One significant contribution of this work is the application of principal component analysis with missing data to object modeling from a sequence of views. An inherent problem in multiple view integration is that the information observed from each view is incomplete and noisy. Based on Wiberg's formulation, we have generalized principal component analysis with missing data as a WLS minimization problem and presented an efficient algorithm, PCAMD, to solve it.

With zero weights assigned to the unobservable data, merging different views can be formulated as a combination of two WLS minimization problems. By applying the PCAMD algorithm to both problems, we get a straightforward two-step algorithm in which the first step computes surface normals and rotation matrices by employing the quaternion representation of the rotation matrix; the subsequent step recovers translation vectors and normal distances to the origin. Experiments on synthetic data and real range images indicate that our approach converges quickly and produces good models even in the presence of noise and mismatching. An accurate polyhedral object model reconstructed from a sequence of real range images is presented. A complex toy house model is also reconstructed.

When motion between two views is relatively small, we can track different segmented surface patches by making use of surface normals, distances, centroids, and adjacency information. An adjacency graph is built for each view and modified as the viewing direction changes. A significant advantage of surface patch tracking, as opposed to other methods such as point matching and line segment tracking, is that surface patches can be more reliably extracted and tracked. One problem in our current system is that the composite graph has to be accurately recovered to extract vertices and edges. This can be done as shown in section 4 and section 5. However, we believe that the composite graph may not be necessary. One possible solution, which we are currently working on, is to combine half-space intersection and union of surface equations and range data to eliminate the use of the composite graph.

Another contribution of this work is a hybrid approach of establishing spatial connectivity of boundary surfaces. The spatial connectivity of surfaces, and in particular, the supporting lines of a simple polygon, can be obtained by combining algebraic equations of surfaces and data points merged from multiple views once transformation is recovered.

"One general principle in computer vision is, if surface information is not enough to determine each surface locally, use global constraints that constrain relative configuration of the surfaces so that the total degrees of freedom decrease." [17]. The object modeling technique presented in this paper is an example of this principle, where the algebraic structure of surface equations from multiple views is used as the global constraint. The recovered object model is statistically optimal because it is most consistent with all of the views in the sense of weighted least squares. By observing and employing different forms of input redundancy, our approach can be easily extended to other vision problems such as shape and motion from a sequence of intensity images. We are also working on applying our techniques to more complicated scene modeling.

Acknowledgment

We thank Sing Bing Kang for his many valuable comments which have significantly improved the quality of this paper. We would also like to thank David Chan for helping segmentation of range images, and Mark Wheeler for proofreading early versions of this paper.

References

[1] N. Ahuja and J. Veenstra. Generating Octrees from Object Silhouettes in Orthographic Views. IEEE Trans. PAMI Vol. 11 No. 2, pp. 137-149, 1989.

[2] F. Arman and J.K. Aggarwal. Model-based Object Recognition in Dense-Range Images - A Review. ACM Computing Surveys. Vol. 25, No. 1, pp. 5-43, 1993

[3] B. Bhanu. Representation and Shape Matching of 3-D Objects. IEEE Trans. PAMI Vol. 6, pp. 340-351, 1984.

[4] Y. Chen and G. Medioni. Object Modeling by Registration of Multiple Range Images. Proc. IEEE Int. Conf. R & A, pp. 2724-2729, April 1991.

[5] C. Debrunner and N. Ahuja. Motion and Structure Factorization and Segmentation of Long Multiple Motion Image Sequences. Proc. ECCV, pp. 217-221, 1992.

[6] D. Dobkin, L. Guibas, J. Hershberger, and J. Snoeyink. An Efficient Algorithm for Finding the CSG Representation of a Simple Polygon. Algorithmica, 10: 1-23, 1993.

[7] Y. Dodge. Analysis of Experiments with Missing Data. Wiley. 1985.

[8] O.D. Faugeras and M. Hebert. The Representation, Recognition, and Localization of 3-D Objects. Int. J. Robotics Research. Vol. 5, No. 3, pp. 27-52, 1986.

[9] F.P. Ferrie and M.D. Levine. Integrating Information from Multiple Views. Proc. IEEE Workshop on Computer Vision. pp. 117-122, 1987.

[10] G.H. Golub and C.F. Van Loan. Matrix Computation. 2nd Edition. The John Hopkins University Press. 1989.

[11] K. Ikeuchi. Generating an Interpretation Tree from a CAD Model for 3D-Object Reconstruction in Bin-Picking. Int. J. Computer Vision. pp. 145-165, 1987.

[12] B. Parvin and G. Medioni. B-rep from Unregistered Multiple Range Images. Proc. IEEE Int. Conf. R & A, pp. 1602-1607, May 1992.

[13] C.J. Poelman and T. Kanade. A Paraperspective Factorization Method for Shape and Motion Recovery. CMU-CS-92-208, Oct 1992.

[14] F.P. Preparata and M.I. Shamos. Computational Geometry. Springer-Verlag. 1988.

[15] A. Ruhe. Numerical Computation of Principal components when Several Observations are Missing. Tech Rep. UMINF-48-74, Dept. Information Processing, Umea Univ., Umea, Sweden, 1974.

[16] M. Soucy and D. Laurendeau. Multi-Resolution Surface Modeling from Multiple Range Views. Proc. IEEE CVPR, pp. 348-353, 1992.

[17] K. Sugihara. Machine Interpretation of Line Drawings. The MIT Press, 1986.

[18] R. Szeliski and S.B. Kang. Recovering 3D Shape and Motion from Image Streams using Non-linear Least Squares. DEC CRL 93/3, 1993.

[19] C. Tomasi and T. Kanade. Shape and Motion from Image Streams under Orthography: A Factorization Method. Int. J. of Computer Vision, 9:2, pp. 137-154, 1992.

[20] S. Ullman. The Interpretation of Visual Motion. The MIT Press. 1979.

[21] B.C. Vemuri and J.K. Aggarwal. 3-D Model Construction from Multiple Views Using Range and Intensity Data. Proc. CVPR, pp. 435-437, 1986.

[22] T. Wiberg. Computation of Principal Components when Data are Missing. Proc. Second Symp. Computational Statistics, pp. 229-236, Berlin, 1976.