Reconstruction of Three-dimensional Objects through Matching of their Parts

Georgios Papaioannou, Member, IEEE, Evaggelia-Aggeliki Karabassi, Member IEEE, Theoharis Theoharis

Abstract. The problem of reassembling an object from its parts or fragments has never been addressed with a unified computational approach, which depends on the pure geometric form of the parts and not application-specific features. We propose a method for the automatic reconstruction of a model based on the geometry of its parts, which may be computer-generated models or range-scanned models. The matching process can benefit from any other external constraint imposed by the specific application.

Index Terms – Object reconstruction, complementary matching, depth buffer, virtual assemblage

1 INTRODUCTION

1.1 Related Work

Up to now the application of pattern recognition and stochastic analysis techniques in three-dimensional computer graphics focuses on object recognition and classification [11], [17], [21], [23], [3]. On the other hand, almost no work has been conducted towards the identification of complementary objects, i.e. objects that can be assembled to form a new solid object. This need arises in various scientific areas, such as computer-aided manufacturing or archaeological applications where the problem is to reconstruct a complete object from its parts.

In most cases the role of a computer in object reconstruction is limited to data acquisition, fragment visualization and manipulation, while the actual reconstruction process is supervised by the scientist [9], [8]. In the two-dimensional case, where the complementary matching is reduced to the “jigsaw puzzle” problem, many solutions have been proposed, as in [7], [6]. To our knowledge, the number of automatic or semiautomatic reconstruction algorithms in three dimensions developed up to now is extremely limited, while the techniques used are applicable only to specific types of object fragments. Some of these algorithms focus on the reconstruction of thin walled object fragments (like pots) and rely either on classification of certain qualitative features of the fragments, as in [19], or comparison of the broken surface boundary curves to match and align the pieces [22]. The first method assumes that the structure of the final, complete object is known a priori and fragments have to be extensively labeled and categorized beforehand. The second addresses the problem as boundary curve matching instead of full surface matching, and therefore cannot handle arbitrary object parts.

Matching algorithms that operate on partial data in three dimensions mostly deal with the fusion of surface segments of the same object rather than perform a complementary matching, and are frequently intended for mesh generation from partial scans, as in [16], [5]. In [1], Barequet and Sharir introduced a robust and noise tolerant method for the matching of point clouds representing the shell or the volume of partially identical objects. This method is based on the geometric hashing paradigm [18], [7] and requires that user defined “footprints” are calculated for or assigned to the evenly distributed points of the data sets.

1.2 Method Overview

The general method proposed in this paper matches and glues fragments or parts belonging to an object, one against one, using only their surface geometry, assuming no information about the fragments’ origin, data set sampling distribution or the final model to be reconstructed. The intrinsic geometric features of 3D objects have been also successfully used in algorithms for other applications like object recognition, as in [3] and [23]. The basic concept in our method is that, given two 3D models, the best fit is likely to occur at their relative pose, which minimizes the point-by-point distance between the mutually visible faces of the objects. For this reason, we introduce and calculate an error measure of the complementary matching between two object parts at a given relative pose, based on this point-by-point distance.

This matching error is minimized, employing a standard global optimization scheme, to determine the relative positioning of the two fragments that corresponds to their best complementary fit. During the automated assembly, it is assumed that the two object parts can be rigidly attached to one another without having to penetrate each other's surface. For instance, the method cannot be used to connect two links of a chain.

In contrast to [22], our method can handle arbitrary objects. It overcomes the constraint of [1] regarding the even spatial distribution of the input samples or the regular topology assumed by [16], by efficiently resampling the data during the evaluation of the matching cost function. In demanding application areas such as the assembling of archaeological finds and simulation, where highly detailed models are used, the regularity of the data sets cannot be guaranteed as the objects are densely sampled only in highly detailed areas, in order to reduce the processing time and storage requirements.

We should note that the proposed scheme may be used without modifications for all types of three-dimensional surface or volume data (analytical surfaces, polygonal models or voxelized objects). However, the implementation of the algorithm is optimized for polygonal data as in this way it takes advantage of commonly available graphics hardware to significantly accelerate the matching.

Due to its generality, our algorithm can work independently, disregarding the morphology and structure of the object to be reconstructed. If additional information is available, e.g. material attributes or structure features, it can be incorporated in the method in the form of a set of constraints, to improve the overall performance.

The paper is organized in the following manner. In section 2 the formulation of the problem is presented. Section 3 provides a theoretical analysis of the proposed matching error and also discusses its practical calculation. Section 4 explains how the matching error is used in conjunction with a global optimization method to solve the reconstruction problem in the general case, while in section 5 we discuss the adaptation of the algorithm to some cases of constrained matching. Finally, in section 6 some representative examples and applications of the method are provided and commented.

2 PARAMETERIZATION OF THE PROBLEM

For two arbitrary objects, assuming a fixed distance between their initial centroids $\bar{O}_1, \bar{O}_2$, one needs to define six degrees of freedom, namely $\theta_1, \varphi_1, \rho_1$ and $\theta_2, \varphi_2, \rho_2$, in order for each model to be able to rotate arbitrarily around the three axes of its local orthogonal reference system (Fig. 1). Rotation angles are mapped to $[0, 2\pi]$. For the comparison of the two objects, surface distances are measured along the direction of the line O_1O_2 crossing the two centroids. In addition to the rotations, the models can slide relative to each other on a separating plane (*p*) perpendicular to O_1O_2. This relative displacement can be effectively modeled by the translation of one of the objects by $\vec{t} = (x_1, y_1, 0)$, assuming that objects are expressed with regard to local coordinate systems aligned with (*p*) as in Fig 1. The models are allowed to move at most by ±10% of the maximum diameter of the both in each direction.

According to the above, we apply one geometric transformation sequence to each object, $\mathbf{M}_1$ and $\mathbf{M}_2$ respectively. These transformation sequences are the combination of a set of rotations, followed by a translation for the first object. Let $\mathbf{R}_{\hat{e},a}$ be the rotation transformation around an axis $\hat{e}$ by *a* radians and $\mathbf{T}_{\vec{V}}$ the translation transformation. If we represent each transformation as a 4×4 homogeneous matrix, as is common in computer graphics and vision [4], a composite transformation can be written as a matrix multiplication. Assuming object points are represented as column vectors, the transformation sequences are: $\mathbf{M}_1 = \mathbf{T}_{\vec{t}}\mathbf{R}_{z,\rho1}\mathbf{R}_{x,\theta1}\mathbf{R}_{y,\varphi1}$ and $\mathbf{M}_2 = \mathbf{R}_{z,\rho2}\mathbf{R}_{x,\theta2}\mathbf{R}_{y,\varphi2}$ (note that the leftmost matrices are applied last). In fact, if the transformations are applied in this order, the rotation $\mathbf{R}_{z,\rho2}$ is redundant and the second transformation sequence can be reduced to $\mathbf{M}_2 = \mathbf{R}_{x,\theta2}\mathbf{R}_{y,\varphi2}$.

As a result, a seven-degrees-of-freedom bounded continuous search space $S^7 \subset \Re^7$ is defined. The set of relative pose parameters form a variable vector in S^7: $\bar{\omega} = [\theta_1, \varphi_1, \rho_1, x_1, y_1, \theta_2, \varphi_2]$.

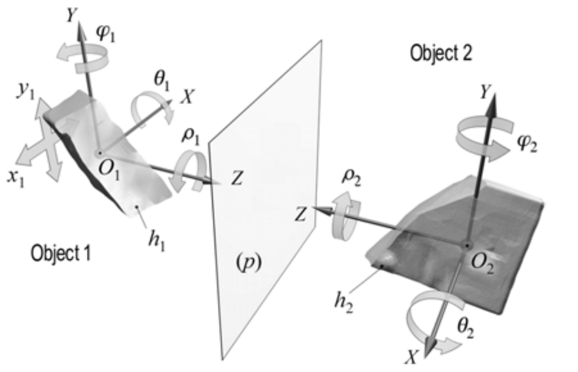


Fig. 1. Relative pose of the two meshes during the matching process.

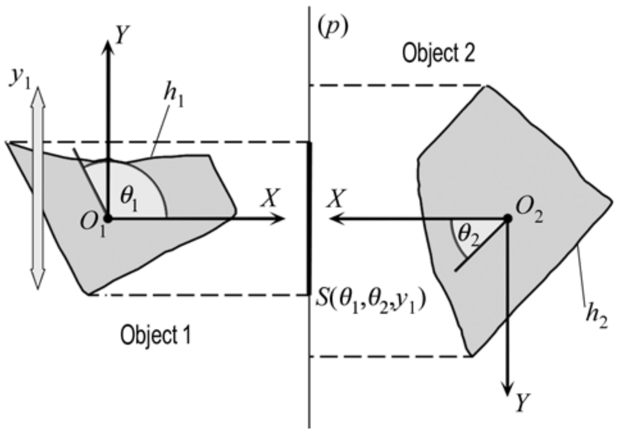


Fig. 2. The simplified two-dimensional case of the complementary matching problem.

3 The Calculation of the Matching Error

In this section the derivation of the matching error $\varepsilon_d(\overline{\varpi})$ is discussed. Our matching method is based on the geometric fitting of two objects, trying to imitate the way a human would attempt to piece together two objects, if no other evidence about the objects is available. The error at every positioning instance $\overline{\varpi}$ is related to the point by point distance of the facing sides of the two objects. For simplicity of presentation, the error calculation for a two-dimensional case will be first derived and then the discussion will be extended to three dimensions.

3.1 The Two-dimensional Case

Let us consider the two-dimensional continuous case shown in Fig. 2. Each object, or rather flat shape, rotates around its center. Additionally, the first object can slide along a path perpendicular to the line crossing the two centers. This way, the vector $\overline{\varpi}$ is reduced to $[\theta_1, y_1, \theta_2]$. The mutually visible profiles of the facing sides of the two objects are represented as $h_1(u,\overline{\varpi})$ and $h_2(u,\overline{\varpi})$. The parameter *u* runs along a separating reference line *(p)* perpendicular to the line crossing the two objects' centers. From Fig. 3a, the distance $d(u,\overline{\varpi})$ between two facing points $h_1(u,\overline{\varpi})$ and $h_2(u,\overline{\varpi})$ on the object contours can be expressed with regard to the respective distances $d_1(u,\overline{\varpi})$ and $d_2(u,\overline{\varpi})$ of these points from the reference line *(p)*:

$$d(u,\overline{\varpi}) = d_1(u,\overline{\varpi}) + d_2(u,\overline{\varpi}) \tag{1}$$

A naïve approach to the error estimation would be to integrate the distance $d(u,\overline{\varpi})$ over the interval $S(\overline{\varpi})$ of *(p)* where the traces of the contours overlap (Fig. 2). However this integration would lead to a dependence of the error on the relative distance between the two objects' centers. Worse, for a certain set of pose parameters at which the object contours would match perfectly, the error estimate could be large due to an improper initial placement of the objects. One such example is presented in Figure 4, where in case (a) the error is larger than the one in case (b), although only case (a) is a perfect match.

In order to eliminate the dependence of the error on the absolute distance between the contours, we could subtract from each measured distance the minimum distance $d_{\min}(\overline{\varpi})$ between the two profiles of the shapes:

$$d(u,\overline{\varpi}) = d_1(u,\overline{\varpi}) + d_2(u,\overline{\varpi}) - d_{\min}(\overline{\varpi}) \tag{2}$$

The corresponding error for a given set of angles $\overline{\varpi} = [\theta_1, \theta_2]$ would be:

$$\varepsilon_d(\overline{\varpi}) = \frac{1}{l} \int_{S(\overline{\varpi})} \left(d_1(u,\overline{\varpi}) + d_2(u,\overline{\varpi}) \right) du - d_{\min}(\overline{\varpi}) \tag{3}$$

where *l* the length of $S(\overline{\varpi})$.

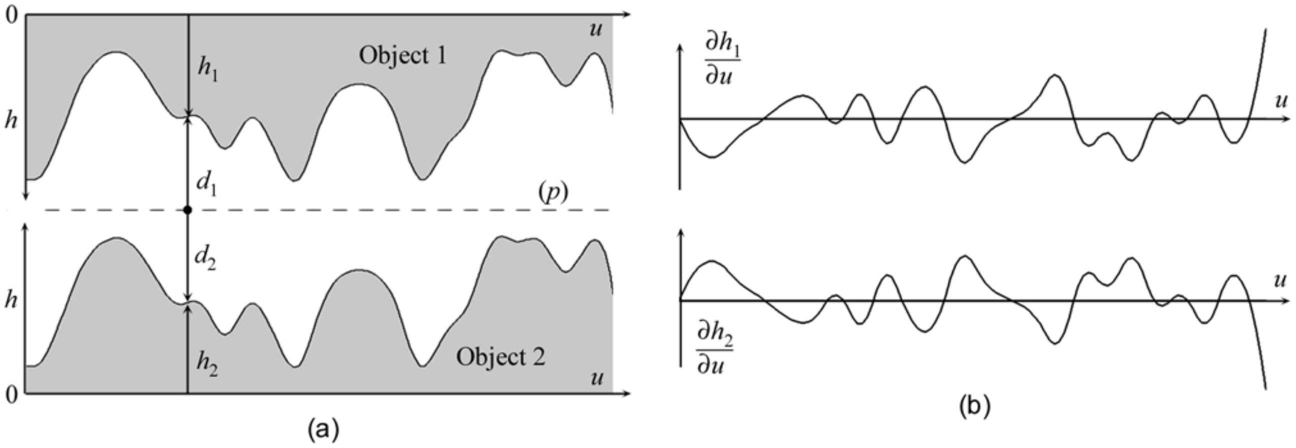

0
h
h
0
Object 1
Object 2
h_1
d_1
d_2
h_2
(p)
u
u
(a)
$\frac{\partial h_1}{\partial u}$
$\frac{\partial h_2}{\partial u}$
u
u
(b)

Fig. 3. Distance estimation in the two-dimensional case. (a) Distance of two object contours relative to a reference plane (*p*). (b) The derivatives of the object contours.

Although this new error estimation is theoretically correct, it is ill-conditioned, as small changes in the object contours (caused for example by the presence of even the slightest jittering of contour points during sampling) affect the minimum distance $d_{\min}(\bar{\varpi})$, which in turn significantly alters the estimated error. To minimise the effect of slight alterations in data to $\varepsilon_d(\bar{\varpi})$ a different approach is necessary which compares the slopes of the contours, i.e. the derivatives of $h_1(u,\bar{\varpi})$ and $h_2(u,\bar{\varpi})$ instead of the actual contour distances. It can be easily seen (Fig. 3b) that matching parts have opposite slopes in corresponding points. The derivatives $\frac{\partial h_1(u,\bar{\varpi})}{\partial u}$ and $\frac{\partial h_2(u,\bar{\varpi})}{\partial u}$ are related to the derivatives of the object contour distances from the reference line (p) as follows (Fig. 3):

$$\frac{\partial h_1(u,\bar{\varpi})}{\partial u} = -\frac{\partial d_1(u,\bar{\varpi})}{\partial u} \text{ and } \frac{\partial h_2(u,\bar{\varpi})}{\partial u} = -\frac{\partial d_2(u,\bar{\varpi})}{\partial u} \tag{4}$$

In other words, the distances $d_1(u,\bar{\varpi})$ and $d_2(u,\bar{\varpi})$ are differentiated with respect to *u* (Fig. 3b) and set the error $\varepsilon_d(\bar{\varpi})$ as:

$$\varepsilon_d(\bar{\varpi}) = \frac{1}{l} \int_{S(\bar{\varpi})} \left| \frac{\partial d_1(u,\bar{\varpi})}{\partial u} + \frac{\partial d_2(u,\bar{\varpi})}{\partial u} \right| du \tag{5}$$

Let one of two nearly perfectly matching surfaces be distorted by a noise spike of length δl and amplitude δd. The limit $\lim_{\delta l \to 0} \delta e_d$ of the additional error δe_d as calculated in (5) is zero as δl decreases, in contrast to $\lim_{\delta l \to 0} \delta e_d = \delta d$ when the direct distance error of eq. (3) is used. The new matching error behaves very well under the presence of noise or mismatching segments as any differences have local effect.

Fig. 5 plots the matching error $\varepsilon_d(\bar{\varpi})$ for the objects shown in Fig. 2. against $\bar{\varpi} = [\theta_1, 0, \theta_2]$. Angles are sampled at 2° intervals. Local minima of the matching error are dense for non-trivial objects, therefore the probability of escaping them using a global optimization method, like simulated annealing, should be high at the beginning of the error minimization process.

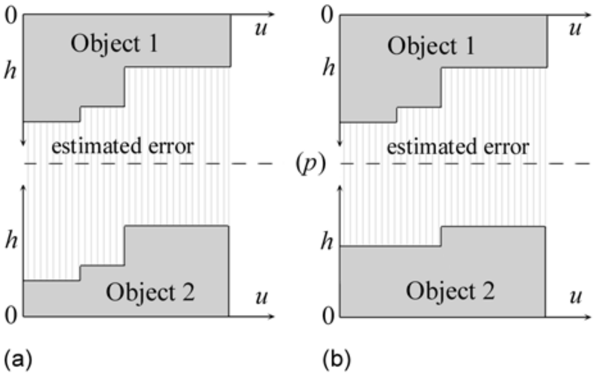


Fig. 4. Naive error estimation in the two-dimensional case. The estimated error for two perfectly matching surfaces (a) can be greater than that for two mismatched surfaces (b).

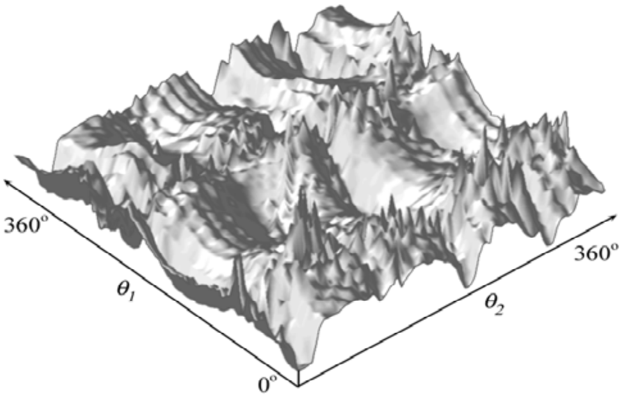


Fig. 5. The matching error for the objects of figure 2 with respect to the rotation of the parts and fixed displacement.

3.2 The Three-dimensional Case

To extend the matching error calculation to three-dimensional space, we need to compute the fitting error as an integral over the slope at every point of the facing surfaces of the two objects. Let $\frac{\partial h(u,v,\varpi)}{\partial u}$ and $\frac{\partial h(u,v,\varpi)}{\partial v}$ be the partial derivatives of an object surface $h(u,v,\varpi)$ with regard to two parameters *u* and *v* respectively, where *u, v* define a plane *(p)* perpendicular to the line crossing the two object centers (Fig. 6). Similar to the two-dimensional case, $\frac{\partial h(u,v,\varpi)}{\partial u}$ and $\frac{\partial h(u,v,\varpi)}{\partial v}$ are related to the partial derivatives $\frac{\partial d(u,v,\varpi)}{\partial u}$, $\frac{\partial d(u,v,\varpi)}{\partial v}$ of the surface distance $d(u,v,\varpi)$ of an object from *(p)* given a pose vector ϖ:

$$\frac{\partial h(u,v,\varpi)}{\partial u} = -\frac{\partial d(u,v,\varpi)}{\partial u}, \qquad \frac{\partial h(u,v,\varpi)}{\partial v} = -\frac{\partial d(u,v,\varpi)}{\partial v} \tag{6}$$

Therefore, by extending the definition of the matching error given in equation (5) to three dimensions we get:

$$\varepsilon_d(\varpi) = \frac{1}{A_s}\iint_{S(\varpi)}\left(\left|\frac{\partial d_1(u,v,\varpi)}{\partial u}+\frac{\partial d_2(u,v,\varpi)}{\partial u}\right|+\left|\frac{\partial d_1(u,v,\varpi)}{\partial v}+\frac{\partial d_2(u,v,\varpi)}{\partial v}\right|\right)dS \tag{7}$$

where ϖ = $[\theta_1,\varphi_1,\rho_1,x_1,y_1,\theta_2,\varphi_2]$, A_s is the area of $S(\varpi)$ and $S(\varpi)$ is the collection of surfaces defined as the intersection of the projections of the two objects $S_1(\varpi)$ and $S_2(\varpi)$ on the plane (*p*), (Fig. 6):

$$S(\varpi) = S_1(\varpi) \cap S_2(\varpi) \tag{8}$$

Note that the quantities $\frac{\partial d_1(u,v,\varpi)}{\partial u}+\frac{\partial d_2(u,v,\varpi)}{\partial u}$ and $\frac{\partial d_1(u,v,\varpi)}{\partial v}+\frac{\partial d_2(u,v,\varpi)}{\partial v}$ are both zero in the case of a point by point perfect match.

3.3 Discrete Error Calculation

For the discretization of the matching error given in (7), the distance between each object and the reference plane (*p*) is uniformly sampled over the areas $S_1(\varpi)$ and $S_2(\varpi)$ for the first and second object respectively. This process can be thought of as casting parallel rays from equidistant locations (*i,j*), *i*=1,…,N_u, *j*=1,…,N_v , lying on a grid on (*p*), towards both objects. Distances $d_1(i,j)$, $d_2(i,j)$ are the distances measured between the ray's origin (*i,j*) on (*p*) and the point of the first hit of the ray on each object (Fig. 7). The intersection of the

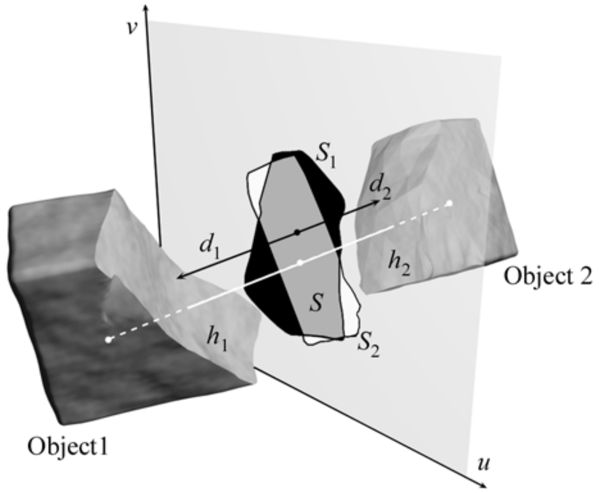


Fig. 6. The error calculation in the 3D case of the matching algorithm.

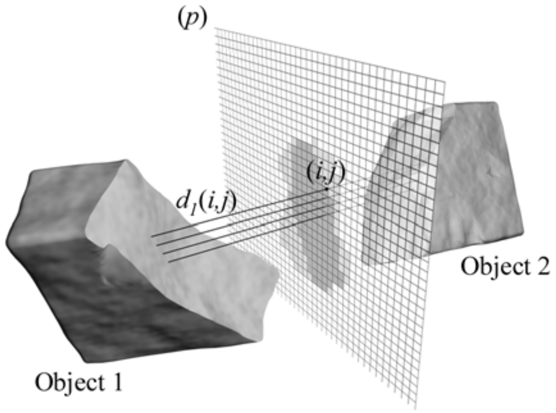


Fig. 7. Discrete error calculation. Object distances from the reference plane (*p*) are estimated using ray casting.

object projection areas $S(\varpi)$ consists of all the locations on plane (*p*) from which cast rays have successfully intersected both objects.

If the partial derivatives $\frac{\partial d_1(u,v,\varpi)}{\partial u}$, $\frac{\partial d_1(u,v,\varpi)}{\partial v}$, $\frac{\partial d_2(u,v,\varpi)}{\partial u}$, $\frac{\partial d_2(u,v,\varpi)}{\partial v}$ of equation (7) are approximated by forward differences

$\Delta_u d_1(i,j) = d_1(i+1,j) - d_1(i,j)$, $\Delta_v d_1(i,j) = d_1(i,j+1) - d_1(i,j)$,

$\Delta_u d_2(i,j) = d_2(i+1,j) - d_2(i,j)$, $\Delta_v d_2(i,j) = d_2(i,j+1) - d_2(i,j)$

and the integration over the area $S(\varpi)$ is replaced by summation, then the error becomes:

$$\varepsilon_d \cong e_d = \frac{1}{N_S} \sum_{(i,j)\in S(\varpi)} \left(\left| \Delta_u d_1(i,j) + \Delta_u d_2(i,j) \right| + \left| \Delta_v d_1(i,j) + \Delta_v d_2(i,j) \right| \right) \tag{9}$$

where $S(\varpi)$={(*i*,*j*) : *i*=1,…,N_u, *j*=1,…,N_v, $d_1(i,j) \neq \infty$, $d_1(i+1,j) \neq \infty$, ,$d_1(i,j+1) \neq \infty$} ∩ {(*i*,*j*) : *i*=1,…,N_u, *j*=1,…,N_v, $d_2(i,j) \neq \infty$, $d_2(i+1,j) \neq \infty$, ,$d_2(i,j+1) \neq \infty$}, allowing only non-infinite values of the forward differences, and N_S is the cardinality of the set $S(\varpi)$.

3.4 Error Calculation using the z-buffer

In most applications where object matching is needed, objects are arbitrarily shaped (e.g. scanned fragments) and can not be approximated by analytical surfaces. In such cases, as in almost every modern three-dimensional computer graphics application, the commonest object representation used is the polygonal mesh. The distance measurement described above incorporates ray casting and intersection tests between every ray cast and every polygon in order to find the intersection point which is closer to the ray's origin; this is a time-consuming calculation in non-convex arbitrary polyhedra.

On the other hand, efficient computer graphics techniques have been developed for the calculation of the distance between the view plane and each visible point on the object surfaces, during the rendering phase. The most widely used algorithm is the z-buffer [2], [4], which is today hardwired even in low-cost home computer graphics boards. The z-buffer algorithm produces a two-dimensional buffer (the z-buffer or depth-buffer), whose dimensions match those of the view plane. Each value of the z-buffer represents the distance between a pixel on the view plane and the corresponding point on the object, which is closest to the view plane. Assuming that the view plane is parallel to the *XY* plane, the z-buffer values are the *Z* coordinates of the object points closest to the view plane.

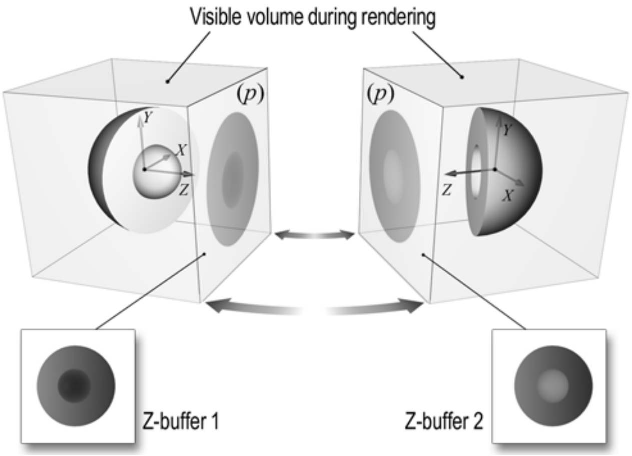


Fig. 8. The use of z-buffer to calculate the point-to-point distances for the matching. The z-buffers for the two objects are shown at the bottom left and right.

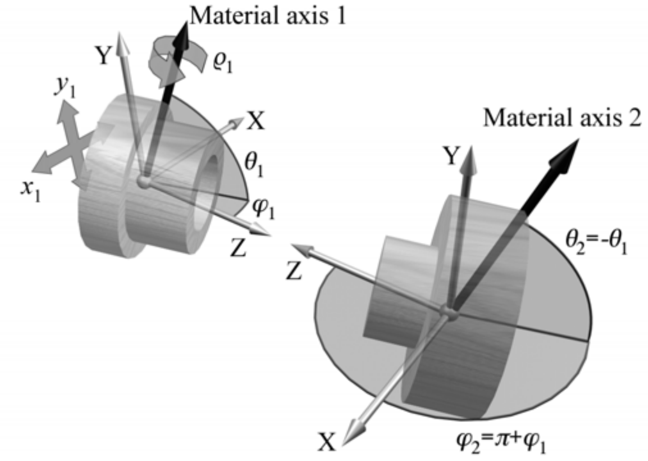


Fig. 9. The use of material axis as a restrictive matching constraint. Objects are shown after rotations around all three axes have been performed.

In our method, we consider the reference plane (*p*) as the view plane and exploit the z-buffer algorithm (and widely available hardware) for the distance calculations. Each object is rendered separately, in a right-handed coordinate system, with the *Z*-axis pointing towards (*p*) (Fig. 8). The resolution $N_u \times N_v$ at which the objects are rendered, represents the coarseness of the discretization approximation (which equals the resolution of the z-buffer and the grid mentioned in paragraph 3.3). As a result we obtain two z-buffers, Z_1 and Z_2, for each rotation vector $\bar{\varpi}$, whose elements correspond to the distances between the reference plane (*p*) and the surface points on objects 1 and 2 respectively. It can be verified from Fig. 8 that the following relations exist between the distance functions and the z-buffers:

$$d_1(i, j) = Z_1(i, j) \tag{10a}$$

$$d_2(i, j) = Z_2(N_u - i, j) \tag{10b}$$

The inversion in the *i* indices of (10b) is due to the fact that the *X*-axes of the two local rendering coordinate systems are looking in opposite directions.

To put together all the above analysis, the practical algorithm for the estimation of the matching error is as follows:

Initialization

1. Calculate the maximum diameter *R* of the two object parts h_1, h_2.
2. Move the parts so that their centers $\bar{O}_1$ and $\bar{O}_2$ reside on the coordinate system origin and rotate them so that the X axes of their local reference frame points in opposite directions:
   $h_1' = \mathbf{R}_{y,-\frac{\pi}{2}} \mathbf{T}_{-\vec{O}_1} h_1$, $h_2' = \mathbf{R}_{y,\frac{\pi}{2}} \mathbf{T}_{-\vec{O}_2} h_2$
3. Prepare the rendering hardware for the acquisition of the $N_u \times N_v$ sized depth images and restrict the viewable area to an $f_R \times f_R \times f_R$ sized cube centered at the coordinate system origin, where $f_R = 1.3 \cdot R$.

Matching error estimation for a pose $\bar{\varpi}$

1. Perform the rigid transformation that defines the given pose $\bar{\varpi}$:
   $h_1'' = \mathbf{T}_{(x_1, y_1, 0)} \mathbf{R}_{z,\rho 1} \mathbf{R}_{x,\theta 1} \mathbf{R}_{y,\varphi 1} h_1'$, $h_2'' = \mathbf{R}_{x,\theta 2} \mathbf{R}_{y,\varphi 2} h_2'$
2. Render the transformed object parts one at a time and store the correponding z-buffer values, $Z_1(i, j)$, $Z_2(i, j)$.
3. Calculate the matching error using equations (9) and (10):

$$\varepsilon_d = \frac{1}{N_S} \sum_{(i,j)\in S(\overline{\omega})} \left(\left|\Delta_u Z_1(i,j) + \Delta_u Z_2(N_u - i, j)\right| + \left|\Delta_v Z_1(i,j) + \Delta_v Z_2(N_u - i, j)\right| \right)$$

The performance of this error estimation technique is extremely high, as all the rendering and z-buffer creation procedures are supported by hardware graphics accelerators, leading to a large number of error calculations per second (approx. 45 matching error values per second, for 64×64 z-buffer resolution, 16-bit accuracy and 10,000 polygons on a Pentium II at 400MHz).

4 MINIMIZATION OF THE MATCHING ERROR

For the determination of the best fit between the objects, we need to find the global matching error. In case of an exhaustive search for the global minimum over the search space, assuming even a coarse quantization (increments of 2^o for each angle and 0.5% for the displacement), more than $75 \cdot 10^{12}$ of error values should be estimated, rendering the approach impractical. Since the exact form of the cost function to be minimized is not known, only a probabilistic cost minimization scheme can be applied.

Any global optimization method can be applied, assuming the cost function to minimize to be the matching error $\varepsilon_d(\overline{\omega})$. The minimization is performed over a seven-degrees-of-freedom search space, the variable vector being $\overline{\omega} = [\theta_1, \varphi_1, \rho_1, x_1, y_1, \theta_2, \varphi_2]$. Rotation angles $(\theta_1, \varphi_1, \rho_1, \theta_2, \varphi_2)$ are wrapped around in the range $[0, 2\pi]$, while x_1 and y_1 are allowed values in $[-0.1 \cdot r, 0.1 \cdot r]$, where *r* is the maximum diameter of both objects.

In our implementation, we adopted a variation of the Simulated Annealing algorithm (SA) [10], the Enhanced Simulated Annealing (ESA) by Siarry et al [20]. In brief, the ESA algorithm is as follows:

```
1. Estimate a good initial temperature $T = T_0$
2. Initialize the parameter vector $\overline{\omega}$ and set $\overline{\omega}_{best} = \overline{\omega}$
3. Calculate the initial value of the cost function $\varepsilon_d = \varepsilon_0$. Set $\varepsilon_{best} = \varepsilon_d$.
4. Randomly perturb the parameter vector $\overline{\omega}$ by $\overline{\delta\omega}$ to get a new vector $\overline{\omega}$ and calculate the new value of the cost function $e_{new}$.
5. If $\Delta\varepsilon = \varepsilon_{new} - \varepsilon_d < 0$ :
        Accept the new vector: $\overline{\omega} = \overline{\omega}_{new}$ and $\varepsilon = \varepsilon_{new}$.
    If $\varepsilon < \varepsilon_{best}$, set: $\overline{\omega}_{best} = \overline{\omega}$ and $\varepsilon_{best} = \varepsilon_d$.
    Else if $\Delta\varepsilon \geq 0$:
        Accept the new vector ($\overline{\omega} = \overline{\omega}_{new}$ and $\varepsilon = \varepsilon_{new}$) with probability $\exp(-\frac{\Delta\varepsilon}{T})$
6. Repeat steps 4-5, until equilibrium is reached at temperature T.
7. Lower the temperature:
        Calculate the average and minimum error values $\varepsilon_{\min}(T)$ and $\varepsilon_{\mathrm{ave}}(T)$ at T.
        Adjust the temperature decrease rate $a = \max\left(\min\left(\varepsilon_{\min}(T)/\varepsilon_{\mathrm{ave}}(T), a_{\max}\right), a_{\min}\right)$.
        Set $T = a \cdot T$.
```

8. Repeat steps 4-7 until T reaches a predetermined minimum value T_{stop} , N_{fail} subsequent temperature states fail to produce a new ε_{best} value or the maximum number of iterations N_{iter} is reached.

9. Return $\overline{\omega}_{best}$.

The above algorithm modifies the well known SA method by introducing an adaptive cooling scheme, a search space partitioning strategy and a Monte Carlo method for estimating the initial and final temperature (convergence control parameters).

The adaptive cooling scheme adjusts the rate of decrease a of the temperature according to the success rate of the previous temperature stage (Step 7). a is initially set to $a_{\min} = 0.6$ while $a_{\max} = 0.9$. The initial (higher) temperature parameter T_0 and the lower one $T_{\min}$ are derived from the Metropolis [12] criterion by performing a fixed number of transitions of increasing cost (50 in our implementation) and setting:

$$T_0 = -\frac{\Delta\varepsilon_{ave}^{(uphill)}}{\ln(P_0)}\ ,\ T_{\min} = -\frac{10^{-6}\cdot\Delta\varepsilon_{ave}^{(uphill)} + 10^{-8}}{\ln(10^{-6}\cdot P_0 + 10^{-8})}$$

where P_0 is the desired initial acceptance probability for uphill transitions and $\Delta\varepsilon_{ave}^{(uphill)}$ is the average increase of the measured cost. The suggested value for P_0 is 0.5. The minimum-cost parameter vector of this stage is used for the initialization of the annealing process.

An important feature of the ESA technique is the search space partitioning. Instead of perturbing all variables $\omega_1,\ldots\omega_N$ of $\overline{\omega}\in S^N$, the transition affects only M of them, $M<<N$. At each transition the M variables are selected by the “least frequently used first” rule. M is fixed throughout the execution of the algorithm. After many experiments we concluded that the best performance for this application was achieved with M=2.

In most implementations, it is assumed that equilibrium is reached at a temperature T if a predefined number $N_{attempted}$ of perturbations has been attempted at T or a predefined number $N_{accepted}$ of perturbed vectors has been accepted at T. In our implementation, we use both criteria for thermal equilibrium and we set $N_{attempted}$ = 80 and $N_{accepted}$ = 8.

For the stopping criterion, we set N_{fail} = 4 and N_{iter}=2000.

Note that the above parameter values were selected after exhaustive experimentation with many test models to achieve as fast convergence as possible. This parameter set worked seamlessly and performed well for all categories of test objects, so no case-specific parameterization was needed. For more ESA implementation details, the interested reader should refer to the original ESA paper.

5. Constrained Matching

Up to this point, we have assumed that both objects are allowed to freely rotate around all three axes, resulting in an unconstrained matching scheme, applicable to all three-dimensional object-matching problems. However in many applications, additional information is available about the objects to be combined. Such information includes material attributes, a priori knowledge of the final reconstructed object shape or constraints in the object matching directions. These features can be used in two ways: to bias the final result or act as restrictive constraints, reducing the search space of the matching algorithm, as will be shown in this section.

5.1 Biasing Constraints

In the case of biasing constraints, the matching process is identical to the one described in the previous sections, but an additional weight is added to the estimated error. For example, certain materials (such as marble or wood) are characterized by veins, which define a “material axis” [15]. The degree of coincidence of the material axes of the two fragments can be used as an additional factor in the error estimation, biasing the result towards pose vectors $\overline{\omega}$ at which the material axes are parallel. The error expression would then become:

$$e = \lambda_d e_d + \lambda_{mat}(1 - |\vec{a}_1 \cdot \vec{a}_2|) \quad (11)$$

In (11) e_d is the matching error from (9), $\vec{a}_1$ and $\vec{a}_2$ are the material axes for each object and λ_d , λ_{mat} are weights depending on the amount of bias desired. Usually, λ_d is close to unit, while $\lambda_{mat} = (1 - \lambda_d)$. In cases of reconstruction of archaeological data with marked marble vein directions, material likeness is an important parameter of the matching process, so λ_d and λ_{mat} must be roughly equal.

Regardless of the contribution of the material factor to the matching error, λ_d can be modified to incorporate a bias towards tighter fitting between the surfaces to be matched. Setting $\lambda_d = \frac{\min(A_{S1}, A_{S2})}{A_S}$, where A_{S1}, A_{S2} are the measured areas of the projected surfaces of h_1, h_2 on (p), the optimization scheme favors solutions of maximum surface overlap between the objects. Notice that if one of the projected areas S_1 or S_2 falls completely inside the other, λ_d =1 and tends to infinity as the two traces get separated. This particular rule for λ_d in conjunction with the directionally restrictive constraint discussed in 5.2.2 has been successfully applied in the reconstruction of scanned objects.

5.2 Restrictive Constraints

Restrictive constraints are used to increase the effectiveness of the matching process by reducing the search space. This is accomplished either by reducing the number of degrees of freedom of the objects or by limiting the range of acceptable values for each state variable.

5.2.1 The material axis as a restrictive constraint

Let us examine the use of the material axis as a restrictive constraint. The demand we can impose on the matching problem is for the two material axes to be parallel at all object poses. The objects should rotate freely around the material axis direction but any other rotation applied to them has to be equal for both parts, in order for the material axes vectors to remain collinear. The matching algorithm can be modified in the following way (Fig. 9):

a) An original orientation of the two objects is selected, which meets the material axes collinearity constraint. For simplicity, the objects are rotated so that the material axes initially coincide with the *Z*-axes of the two local coordinate systems.

b) Rotation angles θ_1, φ_1 and θ_2, φ_2 are locked together so that the material axes remain collinear during this rotations. More specifically, $\theta_2 = -\theta_1$ and $\varphi_2 = \varphi_1 + \pi$.

c) The order of the transformation applied to the first object is altered so that the rotation around *Z*-axis is performed first: $\mathbf{M}_1 = \mathbf{T}_{(x_1, y_1, 0)} \mathbf{R}_{x,\theta 1} \mathbf{R}_{y,\varphi 1} \mathbf{R}_{z,\rho 1}$

By locking the *X* and *Y* rotations for the two parts, we reduce the search space dimension by two, resulting in the variable vector $\bar{\omega} = [\theta_1, \varphi_1, \rho_1, x_1, y_1, \theta_2, \varphi_2] = \left[\theta, \varphi, \rho, x_1, y_1, -\theta, \varphi + \pi\right]$. Of course, the material axis is not always measured accurately and therefore it could be useful to add a small jitter to the θ and φ angles, to allow for a small random deviation from the collinearity condition (±5^o is usually enough).

5.2.2 Fracture direction constraints

Perhaps the most useful constraint, particularly suitable for archaeological fragment reconstruction applications, is the enforcement of a “matching direction” for each object part during the matching process.

Let $\vec{v}_{1m}$ and $\vec{v}_{2n}$, $m = 1, ..., M$, $n = 1, ..., N$, be a set of direction vectors for the first and second fragment respectively, that correspond to the average normal vectors of the fractured faces of each object part. *M* and *N* are the total number of marked directions on the first and the second object part. The search for the best fit should ideally be restricted to those poses at which $\vec{v}_{1m}$ and $\vec{v}_{2n}$ are opposite, modifying the rotation vector

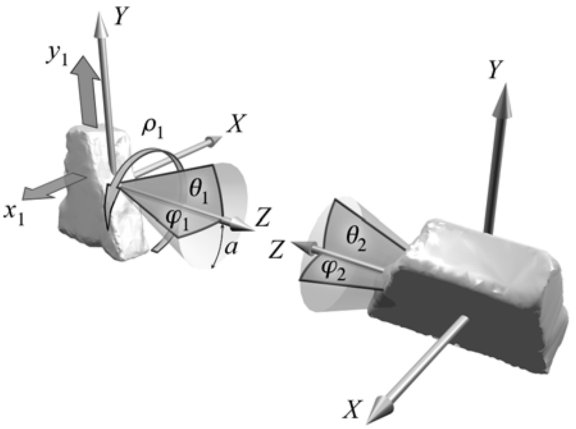
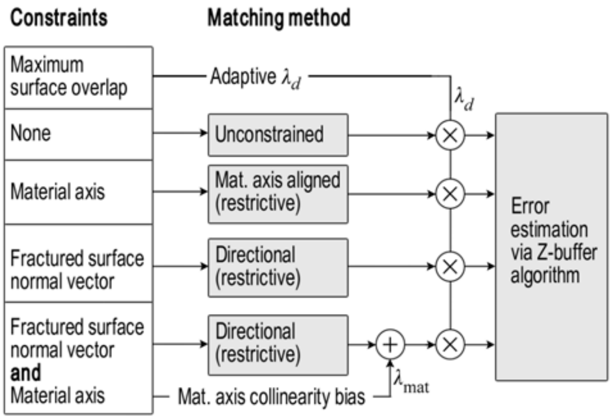


Fig. 10. The object fragments setup for the direction-constrained reconstruction.



Constraints
Matching method
Maximum surface overlap
Adaptive λ_d
λ_d
None
Unconstrained
Material axis
Mat. axis aligned (restrictive)
Fractured surface normal vector
Directional (restrictive)
Fractured surface normal vector **and** Material axis
Directional (restrictive)
Mat. axis collinearity bias
λ_{mat}
Error estimation via Z-buffer algorithm

Fig. 11. The combination chart of the various constraints discussed in this paper.

ϖ to $[0,0,\rho_1,x_1,y_1 0,0,0]$, assuming that $\vec{v}_{1m}$ and $\vec{v}_{2n}$ are aligned with the *Z*-axes. However, since, in realistic applications, directions $\vec{v}_{1m}$ and $\vec{v}_{2n}$ are not likely to be accurately measured, a small deviation from this alignment should be permitted, resulting in the original form for the parameter vector: $\varpi=[\theta_1,\varphi_1,\rho_1,x_1,y_1,\theta_2,\varphi_2]$. Note that this time angles $\theta_1,\varphi_1,\theta_2$ and φ_2 are limited to a small range $[-a,a]$, where a is the *directional tolerance* which depends on the measurement accuracy and is usually less than 10^{o}. Fig. 10 presents the object part configuration for the fracture direction constrained matching.

Taking into account the necessary modifications of the reconstruction method, the algorithm for the direction-constrained matching can be summarized in the following steps:

a) Given the direction vectors $\vec{v}_{1m}$ and $\vec{v}_{2n}$ and the directional tolerance values a_{1m}, a_{2n} for the first and second object part respectively, align each direction vector with the *Z*-axis of the local coordinate system of the object part and choose the axial tolerance a=max{ a_{1m}, a_{2n} }.

b) Maximize the area $S(\varpi)$ by shifting appropriately the first object along the *X* and *Y* axes. This can be done by cross-correlating the area of the projections $S_1(\varpi)$ and $S_2(\varpi)$ on the reference plane (*p*) and finding the position of maximum cross-correlation.

c) Perform the matching error optimization on the set of variables $\{\theta_1,\varphi_1,\rho_1,x_1,y_1,\theta_2,\varphi_2\}$, $\theta_1,\phi_1,\theta_2,\varphi_2\in[-\alpha,\alpha]$.

d) Repeat steps (a) and (b) and (c) for every combination of $\vec{v}_{1m}$ and $\vec{v}_{2n}$, $m=1,...,M$, $n=1,...,N$.

Fig. 11 summarizes the type of constrained matching paradigm to use, according to what constraints are available. Notice that the material axis acts as a restrictive constraint when applied alone, but is used as a biasing constrained if it is combined with another constraint. The bias toward maximum surface can be combined with both the unconstrained and constrained matching cases.

6. Tests and Results

We have tested our reconstruction method on many object models, both computer generated and 3D-digitized models of real object fragments, in various levels of polygonal mesh detail (64 - 25.000 polygons per fragment). Multiple tests were performed with the fragments of all our evaluation objects and the matching error was estimated using z-buffer resolutions between 64×64 and 256×256 pixels, depending on the level of accuracy desired. Larger z-buffer resolutions correspond to longer error estimation time. We found that a good trade-off between accuracy and execution time was achieved for a 128×128 rendering resolution.

6.1 Unconstrained Matching Results

For the unconstrained matching of object parts, we have used both computer generated and digitized polygonal objects. Simple computer generated models with many similar faces (e.g. cubes, pyramids etc) have been avoided, as there is more than one combination of their parts that is an acceptable match.

Fig. 12 displays a list of representative examples of object parts that have been tested with the unconstrained matching algorithm. The first column shows the initial arbitrary (but fixed) position of the parts. The second and third columns present the correct solution $\bar{\varpi}_{corr}$ and the most frequent erroneous result $\bar{\varpi}_{mfe}$ respectively (±5% RMS deviation from $\bar{\varpi}_{corr}$ or $\bar{\varpi}_{mfe}$ is allowed). The average measured error is included in both cases.

Example (a) and (b) demonstrate the effectiveness of the algorithm for both smooth and irregular surfaces. The third example shows a computer-generated ring with cavities where a small ball is supposed to fit. In this example we deliberately chose the worst initial pose where the ball is tested against the void space inside the ring. All experiments resulted in the ball fitting precisely in one of the eight cavities.

In the following three examples we applied the unconstrained matching algorithm on digitized object fragments. The fragments in case (d) belong to a 42×22×11cm ornamental building block. Many minor fragments have been chipped off because of the breaking impact, so the fractured facets cannot perfectly match. The same thing has happened to the fragments of examples (e), (f) where clay pot fragments and pieces of an 11×14×18cm plaster head model have been used. Actually, in application areas like archaeology, a matching algorithm is expected to operate on fragments in bad condition. The tests run with

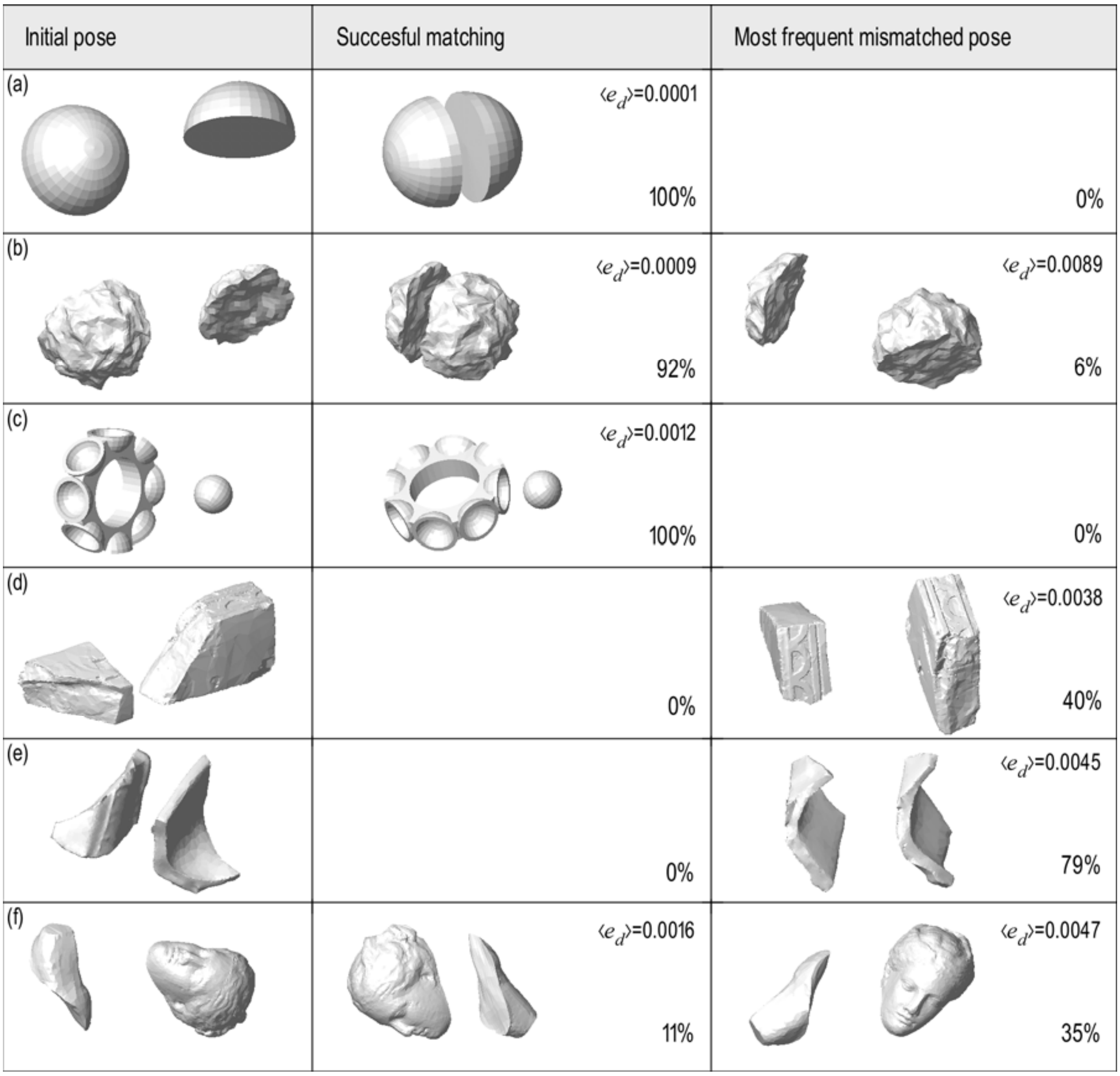


Fig. 12. Reconstruction examples for the unconstrained matching case.

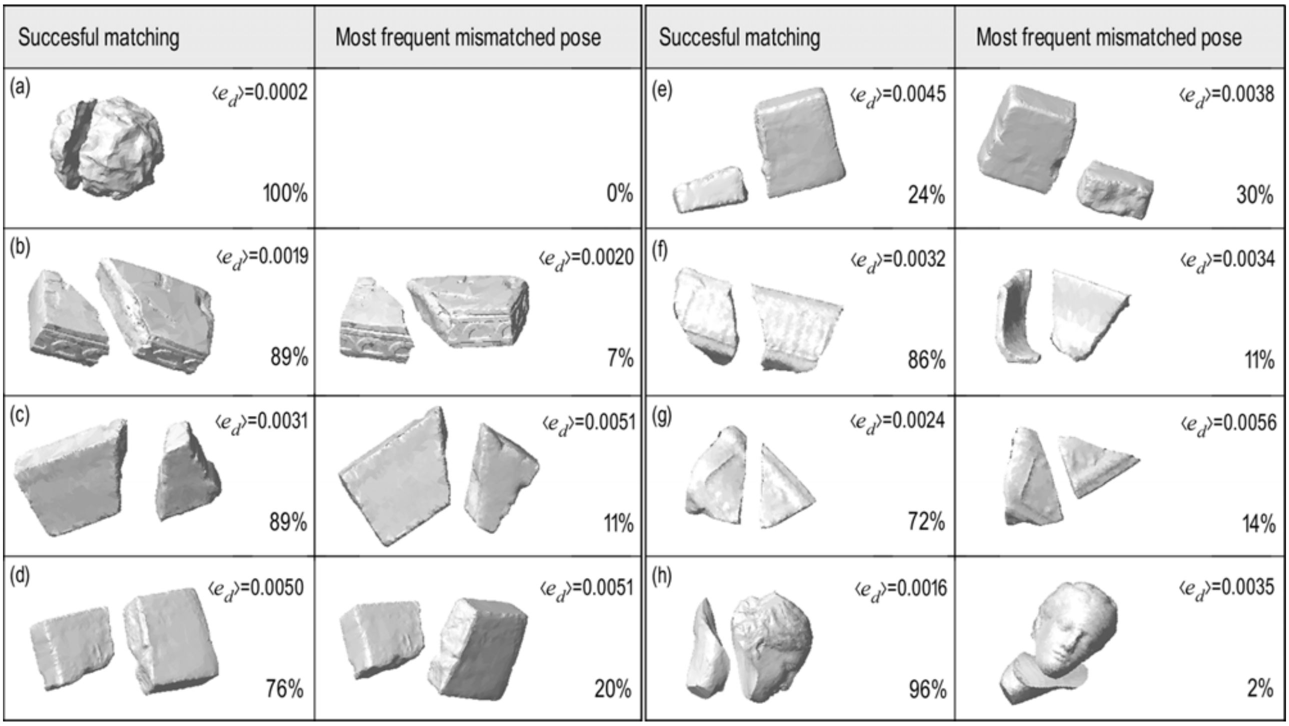


Fig. 13. Representative examples of object reconstruction using the directional constraint and the maximum surface overlap bias.

these fragments, as well as with other similar ones, show that the unconstrained matching favors “trivial solutions” in the case of real objects. Such solutions are relative poses that minimize the matching error between large, smooth portions of the fragments. Although far from the desired results, these trivial solutions are theoretically correct. The unconstrained method has no knowledge to discard a side-by-side match as in case (e). Note also that the probability for a minimization algorithm to converge to a trivial solution becomes higher as the corresponding sides of the fragments become larger and smoother.

Fortunately, real objects frequently have attributes that can be exploited to aid the reconstruction process and therefore, constrained reconstruction is more suitable for them.

6.2 Constrained Matching Results

For the constrained matching tests, we have used the direction-constrained algorithm. The method has been tested mostly with real objects. The directional constraint significantly increases the performance for all types of objects. With the use of an appropriate criterion to restrict the matching to certain directions, the trivial solutions of the unconstrained matching are avoided. Such a criterion is suggested in [14] where arbitrary meshes are segmented into crude compact facets and the algorithm marks as candidate directions for matching the average normal vectors of the most irregular facets. This algorithm is well suited for 3D scanned objects and performs particularly well on types of fragments encountered in archaeological applications.

Fig. 13 presents some examples of directional matching. As the pose parameters are initialized according to one or more predefined candidate directions, the initial pose column of Fig. 12 has no meaning here. The candidate directions were selected using the criterion described above, except in case (a), where a single direction was manually selected.

The first example demonstrates the improvement of the matching performance for the parts of Fig. 12b where unconstrained matching was used. In the second example, 2-3 directions were marked for each fragment, corresponding to the fractured and decorated facets. The facets with the curved designs produced high matching error when examined against other selected sides because they are not complementary with them. As a result, the range of solutions was restricted to those involving the two fractured sides. Most tests were successful although some times the optimization algorithm failed to reach the global error minimum.

For example, in (c), (d) and (e) we used fragments from the same class of flat-surfaced objects but with no decorations. We had the fractured surfaces of these plaster scale models manually punctured and then smoothed out in many places to introduce external error in the process. The pieces in example (d) and (e) come from the same object and we tried to match the smaller ones against the large piece. The smallest piece (case e) failed to produce a correct reconstruction in most cases, as the fracture was badly distorted with respect to its size. In fact, these two fragments had inadequate support even for manual gluing.

In Fig. 13f-g, the matching and gluing between relatively flat pot fragments is demonstrated. The fragments of case (f) are the same we used in the example (e) of Fig. 12. This time, the inner (smooth) walls of the pot are not selected for matching and therefore the trivial solutions are avoided. The results for case (f) are very interesting as both solutions presented here are almost perfect. We had to manually test the actual fragments in order to conclude on which one was the desired solution.

Overall we were satisfied with the results and we were encouraged to apply our complementary matching method to problems where more than one target object must be reconstructed from multiple parts. Such a method is proposed in [13] where the direction-constrained matching and the direction selection criterion of [14] are used for the matching error evaluation between each pair of fragments. Additionally, the maximization of surface overlap and the material axis are used as biasing constraints. A real-time genetic optimization algorithm generates the final part combinations based on the precalculated matching errors and a small but expandable set of rules.

7. Conclusions and Future Work

An innovative method has been presented for the combination of three-dimensional object parts to reconstruct the original objects. The matching algorithm is based on geometric features of the fragments, but it can also exploit additional knowledge; it can therefore be combined with other classification methods. The core of the method is a matching error estimation algorithm based on the distance between the facing fragment sides. When a global optimization algorithm is used with this error estimator as a cost function, the resulting method locates a relative pose of the two object parts at which there is a good complementary match. Our method takes advantage of widely available low-cost hardware to accelerate the process.

We are currently working on the combination of the method proposed here with other, feature-based methods, like the ones listed in the introduction section, in order to investigate the possibility of a reliable and fully automatic object reconstruction procedure.

Acknowledgements

The authors would like to thank Professor Sergios Theodoridis and the reviewers for their valuable comments.

References

[1] G. Barequet and M. Sharir, "Partial Surface and Volume Matching in Three Dimensions", *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 19, no. 9, pp. 930-948, 1997.

[2] E. Catmull, *Subdivision Algorithm for the Display of Curved Surfaces*, Ph.D. Thesis, University of Utah, 1974.

[3] H. Delingette, M. Hebert and K. Ikeuchi, "A Spherical Representation for the Recognition of Curved Objects", *Proc. International Conference on Computer Vision* (*ICCV '93*), pp. 103-112, 1993.

[4] J.D. Foley, A. van Dam, S.K. Feiner, and J.F. Hughes, *Computer Graphics, Principles and Practice*. Reading, Addison-Wesley Publishing Company, pp. 668-972, 1991.

[5] K. Higuchi, M. Hebert and K. Ikeuchi, "Building 3-D Models from Unregistered Range", *Graphical Models and Image Processing*, vol. 57, no. 4, pp. 315-333, 1995.

[6] K. Hori, M. Imai and T. Ogasawara, "Joint Detection for Potsherds of Broken Earthenware", *IEEE Proc. Computer Vision and Pattern Recognition* (*CVPR'99*), pp. 440-445, Colorado, USA, 1999.

[7] A. Kalvin, E. Schonberg, J.T. Schwartz and M. Sharir, "Two-Dimensional, Model-Based, Boundary Matching Using Footprints", *International Journal of Robotics Research*, vol. 5, no. 4, pp. 38-55, 1986.

[8] A.D. Kalvin, T.J. Watson, D. Dean, and J.J. Hublin, "Reconstruction of Human Fossils," *IEEE Computer Graphics and Applications*, vol.15, no. 1, pp. 12-15, Jan. 1995.

[9] A.D. Kalvin, A. Remy, L.J. Castillo, K. Morla, E. Nolasco, J. Prado, V. Fernandez, R. Franco, and G. Wiese, "Computer-Aided Reconstruction of a pre-Inca Temple Ceiling in Peru," *Proc. Computer Applications in Archaeology (CAA 97)*, University of Birmingham, England, April 1997.

[10] S. Kirkpatrick, C.D. Gelatt Jr., and M.P. Vecchi, "Optimization by Simulated Annealing," *Science*, vol. 220, pp. 671-680, 1983.

[11] A.G. Mamistvalov, "n-Dimensional Moment Invariants and Conceptual Mathematical Theory of Recognition n-Dimensional Solids", *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 20, no. 8, Nov. 1998.

[12] N. Metropolis, A.W. Rosenbluth, M.N. Rosenbluth, A.H. Teller, and E. Teller, "Equations of State Calculations by Fast Computing Machines," *Journal of Chemical Physics*, vol. 21, pp. 1087-1092, 1953.

[13] G. Papaioannou, E. A. Karabassi and T. Theoharis, "Automatic Reconstruction of Archaeological Finds – A Graphics Approach", *Proc. International Conference on Computer Graphics and Artificial Intelligence* (*3IA 2000*), pp. 117-125, Limoges, France, 2000.

[14] G. Papaioannou E. A. Karabassi and T. Theoharis, "Segmentation and Surface Characterization of Arbitrary 3D Meshes for Object Reconstruction and Recognition", *IEEE Proc. International Conference on Pattern Recognition* (*ICPR 2000*), Barcelona, Spain, 2000.

[15] G. Papaioannou, T. Theoharis, and A. Boehm, "A Texture Controller," *The Visual Computer*, vol.14 no. 10, pp. 488-496, 1999.

[16] M. Potmesil, "Generating Models of Solid Objects by Matching 3D Surface Segments", *Proc. International Joint Conference on Artificial Intelligence* (*IJCAI '83*), pp. 1089-1093, Karlsruhe, Germany, 1983.

[17] A.P. Reeves, R.J. Procop, S.E. Andrews, and F.P. Kuhl, "Three-Dimensional Shape Analysis Using Moments and Fourier Descriptors," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 10, no. 6, pp. 937-943, Nov. 1988.

[18] I. Rigoutsos and H. Wolfson, "Geometric Hashing", *IEEE Computational Science and Engineering*, vol. 4, no. 4, pp. 10-21, 1997.

[19] R. Sablatnig, C. Menard, and W. Kropatsch, "Classification of Archaeological Fragments Using a Description Language," *Proc. European Signal Processing Conference* (*Eusipco 98*), pp. 1097-1100, Rhodes, Greece, 1998.

[20] P. Siarry, G. Berthiau, F. Durbin and J. Haussy, "Enhanced Simulated Annealing for Globally Minimizing Functions of Many-Continuous Variables", *ACM Trans. on Mathematical Software*, vol. 23, no. 2, pp. 209-228, 1997.

[21] J. Subrahmonia, D.B. Cooper, and D. Keren, "Practical Reliable Bayesian Recognition of 2D and 3D Objects Using Implicit Polynomials and Algebraic Invariants," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 18, no. 5, pp. 505-519, May 1996.

[22] G. Ucoluk and I. H. Toroslu, "Automatic Reconstruction of Broken 3-D Surface Objects", *Computers and Graphics*, vol. 23, no. 4, pp. 573-582, August 1999.

[23] D. Zhang and M. Hebert, "Harmonic Maps and Their Applications in Surface Matching", *IEEE Proc. Conference on Computer Vision and Pattern Recognition* (*CVPR '99*), Colorado, USA, 1999.

Georgios Papaioannou has received his Ph.D. from the Dept. of Informatics and Telecommunications, University of Athens, Greece, in 2001. He received his degree in Computer Science from the same department in 1996. His main research interests are 3D computer graphics, computer vision and pattern recognition. He is currently with the Greek army.

Evaggelia-Aggeliki Karabassi has received her Ph.D. from the Dept. of Informatics and Telecommunications, University of Athens, Greece, in 2001 and her degree in Computer Science from the same department in 1996. Currently, she is working on vision-based metrology systems in Inos Hellas. Her main research interests are 3D computer graphics, computer vision and pattern recognition.

Theoharis Theoharis is an assistant Professor in the Dept. of Informatics, University of Athens, Greece. He received his Ph.D. degree in Computer Graphics and Parallel Processing from the University of Oxford, U.K. He served as a research fellow at the University of Cambridge. His main research interests are in the field of computer graphics and parallel processing.