Image Segmentation and Shape Representation Using Deformable Surfaces[1]

H. Delingette, M. Hebert, K. Ikeuchi
The Robotics Institute
Carnegie Mellon University
5000 Forbes Avenue, Pittsburgh PA 15213

Abstract

We present a technique for constructing shape representation from images using free-form deformable surfaces. An object is modelled as a closed surface that is deformed subject to attractive fields generated by input data points and features. Segmentation is achieved by initializing the surface at some location in the scene and by letting it deforms itself until it fits the input data. Surface deformation is controlled by applying standard theory of mechanical systems. The algorithm is general in that it makes few assumptions on the type of features, the nature of the data and the type of objects. The algorithm does not assume that a clean segmentation of the input data is available. It correctly recovers shape even in the presence of spurious features and data points. We present results in a wide range of applications: reconstruction of smooth isolated objects such as human faces, reconstruction of structured objects such as polyhedra, and segmentation of complex scenes with mutually occluding objects. We have tested the algorithm using data from different sensors including grey-coding and laser range finders and video cameras, using one or several images. We briefly describe the application of this representation to the problem of computing stable grasp position for the manipulation of unstructured objects.

1 Introduction

The recovery of object shape from 3D data is one of the key issue in vision. One could define this task as the segmentation of a large set of data points into shapes corresponding to objects in the scene. The shape representation should be general enough to handle a wide variety of scenes yet simple enough to be usable for other tasks such as recognition and manipulation. In other words, the shape representation should have enough parameters to describe the specificity of

1. This research was supported in part by NASA under Grant NAGW 1175, and in part by DARPA through ARPA order No. 4976. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied of NASA, DARPA, or the US government.

the shape but must have as few parameters as possible to be usable and to be robustly extracted from visual data. This conflict is related to the scale space problem, where one would like to find a description fine enough to capture the key details of the shape, but coarse enough to get rid of spurious details that would only penalize applications that use the description, and that would render the shape extraction process less robust.

In this paper, we propose an approach that attempts to resolve this conflict by substituting to fine / coarse opposition, the feature / data opposition. Several psychophysical experiments have proved that the human eye is able to capture the main shape of an object by seeing only a few characteristic elements or features. These features can be either geometric (distance discontinuities, surface orientation discontinuities, corners, minimum of curvature, etc.) or higher level such as reflectance properties. While these features capture most of the shape information, it is difficult without apriori knowledge to construct a full reconstruction of the object.

Several solutions have been proposed. Besl and Jain[6] built curvature-based object representations by classifying surfaces according to the sign of its principal curvatures. Pentland[22] presented a physically-based algorithm, to recover in a unique manner a model from a set of features and a set of vibration modes. In this approach, the key is to find the correct features, thus restraining the algorithm to either smooth or structured shapes. Another approach is to use both features and range data in separate stages. In a first stage, features are grouped into hierarchical sets, according to geometric properties (symmetry, connexity, etc.) and in the second stage models are fit to the segmented parts This idea of hierarchical representation was initiated by Marr and Nishihara[22] and pushed further by the seminal ACRONYM[3] vision system by using generalized cylinders. Pentland[14,15]‘s “Representation by parts” using deformed superquadrics, proved to have some successful results but encounters some limitations. While the feature grouping requires some accurate feature extraction and high level reasoning, the fitting of superquadrics[21,22,14] to the range data has some unstable behavior, due to its non-linear nature, and is suitable for only smooth and simple shapes.

Those techniques attempt to represent all shapes by using a set of elementary shapes (superquadrics, generalized cylinders, parametric patches, etc.) that can be described by a few parameters. This is clearly beneficial from the point of view of object recognition which amounts to manipulating analytical equations of the elementary shapes. In practice, however, it restricts considerably the class of objects and scenes to which the techniques can be applied. The shape descriptions will always be rough approximations unless the object being modeled can be exactly described by one of the elementary shapes. More general representations could be obtained by adding degrees of freedom to the elementary shapes (e.g., adding tapering and bending to superquadrics). However, the non-linear fitting algorithms involved in the recovery of such shapes become rapidly computationally expensive and numerically unstable. Furthermore, most of those techniques assume that the observed scene is first segmented in regions corresponding to individual objects. The shape extraction algorithms are then applied to each object. However, accurately segmenting a scene is a hard problem in itself. Therefore any shape representation algorithm that requires a perfect segmentation is doomed to fail in realistic situations.

To address those problems, Terzopoulos and Witkin[22, 16 , 22] proposed the concept of deformable contours and deformable surfaces. In this approach, contours or surfaces are deformed under the influence of forces generated by image elements such as edgels. Deformable contours are used for extracting 2-D shapes, deformable surfaces are used for extracting 3-D shapes from range and intensity images. In the latter case, for example, they used a tube made of elastic mate-

rial that was deformed by a potential field computed from an intensity image. Using symmetry and smoothness as internal constraints, this tube is able to approximate the main shape of the object. But to perform correctly, the tube had to be initialized close to the line of symmetry because the deformations occur only locally. This work demonstrated that deformable shapes are powerful tools for image segmentation and shape representation.

Our approach is also to represent shapes by deformable surfaces. We consider surfaces that are topologically equivalent to the sphere. Given an observed scene, a surface is initialized in the vicinity of detected features. The surface is then deformed subject to forces generated by features and data points. The forces generated by data points control *local* shape, while forces generated by features control *global* shape. A smoothness energy is added to the deformation equations to take into account the fact that data and features may be sparse and noisy. Clearly, such free-form surfaces can represent a large class of objects since few constraints are put on the resulting shape. However, our aim is also to build a shape extraction algorithm that does not depend on the nature of features and data. Furthermore, the algorithm must be robust enough so that it does not require a precise segmentation of the scene as an input. The deformable surfaces algorithm presented in this paper satisfies those goals. Specifically, it has the following characteristics:

• **Physically based algorithm:** The representation is obtained by deforming a surface. The dynamics of the deformation is modelled by the Lagrangian equations of mechanical systems. The surface is constrained by external actions from features and range data and by internal actions. As demonstrated in previous works, using physically based procedures provides better control on the stability and convergence of the algorithm.

• **Optimal shape description:** Different pieces of information can be given different weights in the algorithm. For example, a strong feature like a corner would have more influence on surface deformation than data points measured in a featureless area. This leads to an optimal shape description in that it realizes the best compromise between different shape clues such as different types of features and data points with varying levels of noise.

• **Stability:** because the algorithm uses both features and data, it is less sensitive to spurious features, noisy data or missing data. Moreover, this stability enables to perform segmentation by discarding the features and data that is incompatible with the current shape of the surface. Therefore, our algorithm does not require a clean segmentation of the scene, an approximate partition of data and features into region of interest is sufficient. This is in sharp contrast with other techniques that require that the observed scene is already segmented into regions that correspond exactly to individual objects. Furthermore, the stability of the algorithm allows us to deal with non-uniform data distribution and heterogeneous features.

• **Generality:** The algorithm makes few assumptions on data and observed objects. The only requirement is that some features can be extracted from input data and that the minimum and maximum sizes of the object expected in a typical scene are known. We have applied successfully the algorithm to the reconstruction of smooth isolated objects such as human faces, to the reconstruction of structured objects such as polyhedra, and to the segmentation of complex scenes with mutually occluding objects. We have tested the algorithm using data from different sensors including grey-coding and laser range finders and video cameras, using one or several images. The generality of the algorithm

allows us to deal with this array of applications without modifying its core. All that needs to be changed is the front-end program that converts feature and data from the sensor to an internal data structure.

Section 2 describes the theory of deformable surfaces, Section 3 describes the implementation for various sensors and applications.

2 Deformable surfaces

The basic surface description is a free-form deformable surface. The surface is subject to a field of forces resulting in a total surface energy. The energy changes over time as the surface is deformed. A deformable surface is subject to three types of forces, external, internal, and inertial forces.

External forces are generated by input data points and input features. External forces apply deformations that bring the surface as close as possible to the data. External forces must be carefully designed to obtain an optimal shape description that uses features to constrain global shape and data points to fine-tune local shape (Figure 2). External forces can be represented as links between the surface and data points and features (Figure 2). Earlier work on surface reconstruction using splines subject to constraints from data points required a complete mapping between data and surface points to be defined beforehand (see Figure 1). This constraint restricts the geometry of the problem and requires the surface to be initially very close to the solution. To remove those constraints, we need to design the forces so that there is no restriction on their geometry.

Internal forces are generated by the surface itself as it is deformed. The inclusion of internal forces ensures that the surfaces will not tear apart, fold onto itself, or exhibit high curvature points or sharp discontinuities in curvature. The other role of internal forces is to provide constraints in regions in which little or no data is available. This is similar to the regularization approach to surface reconstruction from sparse data[17,22,22]. The standard way of defining internal forces is to define the corresponding energy as the integral over the surface of the magnitude of the first and second derivatives[16,22,22,22] which characterizes the surface smoothness. The relative importance of external and internal forces is a trade-off between accuracy and smoothness. High internal forces generate a very smooth surface that may be far from the input data. Low internal forces allow the surface to fit the data closely but they also allow the surface to fit any noise in the data. Furthermore, they do a poor job at interpolating regions with sparse data.

Inertial forces are generated by the motion of the surface as it evolves over time assuming that the surface has a non-zero mass. Inertial forces are necessary to model the deformable surface as a dynamic mechanical system.

In this section we first describe the components of the three types of energy and the equations of motion (Section 2.1), then we describe in detail how each energy is computed (Sections 2.2, 2.3, 2.4) in the case of a continuous deformable surface. In Section 2.5 we describe the computation of the energies and the implementation of the equations of motion in the case of a discrete deformable surface.

2.1 General equations of motion.

The internal and external energies involved in the deformation of the surface are:

• **Smoothness energy $E_{smoothness}$**: The smoothness energy is a measure of the average curvature of the surface. It is necessary to guarantee that the surface is reasonably

smooth, especially in the case of sparse data. The smoothness energy is internal in that it depends only on the shape of the surface in the vicinity of each point.

• **Feature energy $\mathbf{E_{feature}}$**: The feature energy quantifies the effect of the features on the surface. Each surface point is attracted by each feature. The magnitude of the attractive field is function of the distance between surface point and feature.

• **Data energy $\mathbf{E_{data}}$**: The data energy quantifies the effect of data points on the surface. Theoretically, each surface point is subject to an attractive force by each data point. However, in practice the closest data point is the only one taken into consideration.

To calculate the equilibrium position of the surface using mechanical systems theory, we need to introduce two additional inertial energy terms:

• **Kinetic energy T**: A mass μ is associated with each data point thus generating a kinetic energy term. Unlike other dynamic splines[16,22], our scheme is explicitly linking the deformation of the surface with the minimization of the potential energy.

• **Raleigh Dissipation energy D**: The dissipation term is added to simulate the exchange of energy between the dynamic surface and a virtual medium in which it evolves. This damping term is added to avoid cases in which the surface oscillates forever around an equilibrium position.

Following the equations of mechanics and the principle of least action[21], the surface reaches a stable equilibrium when the Lagrangian of the system of forces reaches a minimum. The Lagrangian of the system is: $\mathbf{L = T - D - E_{smoothness} - E_{feature} - E_{data}}$. A deformable surface is parametrized by two spatial coordinates *(u,v)* and by the time *t*. A surface point P has coordinates:

$$\vec{r}(u,v,t) = \begin{bmatrix} X(u,v,t) \\ Y(u,v,t) \\ Z(u,v,t) \end{bmatrix}$$

With these notations, using the calculus of variations on *L(u,v,t)*, one gets:

$$\mu \times \frac{\partial^2}{\partial t}\vec{r}(u,v,t) = -k \times \frac{\partial}{\partial t}\vec{r}(u,v,t) + (\vec{F}_{smoothness} + \vec{F}_{feature} + \vec{F}_{data})$$

where μ is the mass density of the surface and *k* is the damping factor.

This surface is being deformed between $t=0$ and $t=T_0$. Thus, unlike other scheme using deformable splines[22,16,22,16,22,22] in which the surface is deformed until an error is less than a threshold, we control the amount of deformation of the surface by imposing a time limit T_0. This allows us to deal with noisy and perturbing data.

MORE ON THE MYSTERIOUS T0

2.2 Smoothness Energy

We use the standard Tikonov's stabilizers of the first and second order as a smoothness measure. The first order measures the distance discontinuities while the second order measures the surface orientation discontinuities. Denoting partial derivatives by subscripts (e.g., $\vec{r}_u = \frac{\partial}{\partial u}\vec{r}(u,v,t)$, the energy is defined by:

$$E_{smoothness} = \int_0^{T_0} \left(\iint_{0\,0}^{1\,1} [\alpha_1 \times (\|\dot{r}_u\|^2 + \|\dot{r}_v\|^2) + \alpha_2 \times (\|\dot{r}_{uu}\|^2 + 2\|\dot{r}_{uv}\|^2 + \|\dot{r}_{vv}\|^2)]\, du \times dv \right) dt$$

The corresponding force is therefore:

$$\vec{F}_{smoothness} = \alpha_1 \times \begin{bmatrix} \Delta_{uv} X \\ \Delta_{uv} Y \\ \Delta_{uv} Z \end{bmatrix} + \alpha_2 \times \begin{bmatrix} \Delta_{uv}\Delta_{uv} X \\ \Delta_{uv}\Delta_{uv} Y \\ \Delta_{uv}\Delta_{uv} Z \end{bmatrix}$$

The coefficients α_1 and α_2 controls the relative smoothness of the surface.

2.3 Feature Energy

Because every feature contributes to the global deformation of the surface, our approach is to link every feature to every point of the surface. The links can be seen as springs attached to each feature. Feature energy is given at a surface point by the sum of the energies generated by all the springs attached to the features (see Figure 3):

$$E_{feature} = \sum_{i=0}^{n} E^i_{feature}$$

With this definition of feature forces, the algorithm satisfies two requirements:

- No initial mapping is needed, so that the initial surface can be “far” from the object.
- Due to the springs actions, the points are going to concentrate toward the features, thus yielding an optimal global description.

To get both global and local deformations is hard because the two types of deformation, feature and data, should be balanced. If only feature deformation were applied one would get a surface that connects the features with mostly planar surfaces in between because of the smoothness constraint. On the contrary, in the case of data deformation, one would get a surface that would faithfully follows the shape of the object only where the surface is close enough initially(see Figure 4). Our solution is to change the relative influence of both types of deformations over time: initially, the surface is mostly submitted to feature forces, putting the surface close to object; then the feature forces decrease to be replaced by data forces. This amounts to decreasing over time the distance within which the points of the surface are attracted by features. In the current implementation, we model features as 3-D line segments. We denote the midpoint of the segment by F_i, and its length by l_i. With these notations, the energy field generated by feature *i*, $E^i_{feature}$ is:

$$E^i_{feature} = \int_0^{T_0} \left[\iint_{u\,v} U\left(\frac{l_i \|P\vec{F}_i\|}{D_0^i \times (t_0 - t)} \right) du \times dv \right] dt$$

where:

- *U(x)* is a function that is quadratic if *x<1* and constant if *x>1*. *U* becomes constant after *x = 1* to avoid large attractive forces generated by features that are very far away.

This improves the robustness of the algorithm since spurious features that are not part of the object will not contribute significantly to the forces.

• $l_i\|\vec{PF}_i\|$ is an approximation of the integral of the Euclidean distance between $\vec{r}$ and the feature points over the segment, normalized by the length of the segment. Other feature representations can be used in the same framework by replacing this term by the appropriate value. For example, for a point feature it would be the distance between surface point and feature point.

•s D_0^i is the distance between the feature and the center of the surface at its initial position. If the surface is initialized as a sphere, it is the distance to the center of the sphere.

Figure 5 shows how the potential field varies over time in the case of four coplanar features. The center of the surface coincides with the center of gravity of the features in this example. This example demonstrates that this dynamic potential avoids the "local minimum" problem, because there is no fixed valleys in which points can be trapped.

From the definition of feature energy, the force generated by the feature *i* at surface point *P* at time *t* is:

$$\vec{F}^{\,i}_{feature} = \frac{l_i}{D_0 \times (T_0 - t) \times \|\vec{PF}_i\|} \times \frac{d}{dx}\left(U\left(\frac{l_i\|\vec{PF}_i\|}{D^i{}_0 \times (T_0 - t)}\right)\right) \times \vec{PF}_i$$

This force is roughly equivalent to the force of a spring if $l_i\|\vec{PF}_i\|$ is less that $D_0^i \times (T_0 - t)$ and is null otherwise. Forces from each feature may be weighted to reflect the relative importance of different types of features.

2.4 Data Energy

Since data information is used only for local deformation, the corresponding force should decrease sharply with distance. Therefore, a gravity-type field instead of a spring model is appropriate for this type of deformation. Theoretically, a surface point is subject to forces from all the data points. However, for computational reasons we take into account only the closest data point. For every point *P* of the surface, the closest data point is denoted by P_{data}. If *W* is the gravity potential, the resulting data energy is given by:

$$E_{data} = \int_0^{T_0}\left[\iint_{u\,v} W\left(\frac{\|\vec{P}P_{data}\|}{K}\right)du \times dv\right]dt$$

Where *K* is a normalizing constant that has the dimension of a distance. Intuitively, surface points are influence only by data points that are at a distance less than *K*. Choosing *K* small enough compared to the maximum expected object size guarantees the locality of forces generated by data points. A gravity potential should be in 1/distance everywhere. To avoid infinite potential when the distance vanishes, the potential becomes a quadratic function near 0 (see Figure 7). The corresponding force is:

$$\vec{F}_{data} = \frac{W'\left(\frac{\|PP_{data}\|}{K}\right)}{\|\vec{P}P_{data}\|K} \times \vec{P}P_{data}$$

2.5 Implementation

We have assumed so far that our model is a continuous surface topologically equivalent to a sphere parameterized in *(u,v,t)*. In practice, however, we can manipulate only discrete surfaces. This raises the problem of the parameterization of such surfaces, and, in particular, the impossibility to map a sphere into a square in a uniform way. To avoid creating poles, we adopt the tessellated icosahedron as a structure. Each face of the icosahedron is subdivided to yield arbitrary resolution of the parameter space. The number of faces of the tessellation is $20N^2$, where N is the frequency of the subdivision. Typically, we use $N=5$ yielding a decomposition of the parameter space into 500 faces. Figure 8 shows the tessellation of the sphere for $N=5$. We use the center of each triangle as a node, every node having therefore three neighbors. The time axis is also discretized. The deformable surface is updated at unit time increments. If we write $\vec{X}_t^i$ as the position of the node i at the time t, then the discrete version of the motion equation is:

$$\vec{X}_{t+1}^i = \vec{X}_t^i + (1-k) \times (\vec{X}_t^i - \vec{X}_{t-1}^i) + \vec{F}_{smoothness} + \vec{F}_{data} + \sum_{i=0}^{n} \vec{F}_{Feature}^i$$

The surface is initialized as a sphere at $t=0$. The surface is deformed by applying repeatedly the equation of motion at each node. Forces $\vec{F}_{data}$ and $\vec{F}_{feature}^i$ are computed at each node independently in a parallel manner. $\vec{F}_{smoothness}$ is computed by approximating the first and second derivatives of the surface by finite differences. The computation of $\vec{F}_{smoothness}$ can also be carried out at each node independently except that it involves neighboring nodes. The most expensive part of the algorithm is the computation of the closest data point P_{data} used in the computation of $\vec{F}_{data}$. It is theoretically in $O(lm)$ where l is the number of data points and m the number on nodes. In practice, some precomputations and assumption allow to improve greatly this computational time. The preprocessing depends on the nature of the data and is described in the next Section for each experiment. This preprocessing does not affect the deformable surfaces algorithm itself since it is part of the front end data preparation. The algorithm is otherwise linear in the number of features and the number of iterations. Due to its highly parallelizable nature, substantial speed-ups can be achieved.

Several parameters must be set to apply the motion equation. It is important to note that most parameters are independent of the nature of the data. The parameter settings in the current implementation are:

- **Radius and center of initial sphere:** The determination of center and radius of the initial sphere depend on the nature of the data. The initialization procedure is described in the next section for each type of data.

- **Number of iterations, T_0:** The number of iteration is T_0 (between *0* and T_0 *-1*) and is related to how far the initial shape is to the object; the largest T_0 is and the smallest the deformation is between two iterations. We currently use $T_0 = 200$.

- **Smoothness coefficient α_1 and α_2:** The smoothness coefficients should be between 0 and 1 with $\alpha_1 > \alpha_2$. Actual "good" values have to be determined empirically. We use $\alpha_1 = 0{,}75$ and $\alpha_2 = 0{,}4$ in our experiments.

- **Damping factor *k*:** The damping factor should be close to (but lower than) 1 to ensure smooth deformation of the surface over time and to avoid oscillations. We use $k = 0{,}9$.

- **Normalizing factor *K:*** K depends on the environment. It is computed as one-fifth of

the radius of the largest object expected in a scene for a given application. The algorithm is robust with respect to K so that a rough estimate of object size is sufficient.

3 Experimental Results

In this section we present experimental results obtained by applying the deformable surface algorithm to real image data. We present two sets of results. The first set is obtained using range data from an light-stripe range finder. Those experiments show that the algorithm can extract 3-D object shape from range images without requiring good feature extraction or perfect scene segmentation. The second set of experiments involves applying the algorithm to intensity images. The goal of those experiments is to validate the claim that the algorithm is independent of the nature of the data. We show that the algorithm correctly reconstructs shapes from intensity images.

3.1 Range Data

For our experiments with range data, we use a commercial light-stripe range-finder that consists of a camera and a projector that projects patterns commanded by a LCD board[22]. The sensor processes the images of the patterns using standard light-striping geometry. It delivers a set of four images of 240 rows by 256 columns, one intensity image and three images of the three coordinates of every pixel with respect to a reference frame. Several sensors can be used at once to yield multiple views of a scene. A calibration procedure is used to express all data points coordinates with respect to a single world-centered frame. We conducted the experiments with either single views or multiple views. Using multiple views demonstrates that the algorithm is completely independent of an image-centered reference frame. In particular, it does not use the grid structure of the image or the uniform sampling of pixels in the image. This shows that it can be used with non-imaging sensors that measure non-uniform sparse data. This is a major difference with other reconstruction algorithms that use the image grid as a discretization of the surface.

We analyze the experiments in a simple to complex fashion. We first discuss two experiments in which images of isolated objects, a cube and a human face, are used. Those two experiments are intended to show the basic operation of the algorithm. The choice of the two objects is not arbitrary. The example of the cube shows that the algorithm can recover the shape of objects with corners. At the other extreme, the example of the human face shows that the algorithm can recover the shape of a complex smooth object, including small local shape features. We then describe experiments in which scenes with multiple occluding objects are segmented and the object shapes are recovered. We use two examples. One is the segmentation of natural scenes that contain rocks and pebbles. The second one is the segmentation of a scene with three cubes stacked together. We use simple known shapes in the second example to be able to evaluate the quality of segmentation and shape recovery. In those two examples, no segmentation is applied prior to applying the deformable surface algorithm. Only a rough guess of where the objects are in the scene in needed. This demonstrates that the algorithm performs image performs scene segmentation as well as surface reconstruction.

Isolated Cube

Figure 9.a,b shows two range images of a cube that have been taken from opposite directions. Two types of features are computed, distance discontinuities and surface orientation discontinuities by using standard edge operators. The edge magnitude image is thresholded and a thinning algorithm is applied to the resulting regions. After polygonal approximation, the final feature representation is a set of 3-D segments. Figure 9.d shows the segments. Because of the geometric nature of the object, we are able here to extract correctly most of the edges of the cube.

The deformable surface is initialized as a sphere (see Figure 9.e); the sphere is set so that one diameter passes through a given point and is parallel to the line of sight of the camera (Figure 11). This procedure has the advantage of requiring only the coordinates of one point (the line of sight is given by a calibration file), thus allowing a fully automatic procedure if a high-level module can determine such point.

The result of the deformation is shown in Figure 9.f and Figure 10. Figure 10 shows the triangulated surface (every node has three neighbors). Figure 9.f gives a synthetic aspect of the reconstructed object using ray-tracing. It can be seen that the algorithm correctly recovered sharp edges and corners, even though they are slightly rounded due to the smoothness constraint. This is due to the optimal description capability of our algorithm: it differentiates information from features information from range data.Figure 10 shows clearly that those edges correspond to a concentration of nodes.

Human Face

A human face is a good example of a complex object with parts of very distinct nature: the forehead and jaws areas are of little interest for face recognition, whereas eyes, nose, mouth and chin are the main characteristics of a face. Building compact and accurate surface representation is crucial in the area of face recognition[4]. We would like a surface reconstruction algorithm to smooth the input data and generate a compact representation of the face while keeping an accurate description of the areas of interest. We applied our algorithm to a range image of a human face to verify that it does perform as desired. Figure 12.a shows the intensity image and Figure 12.c shows the range data of a face, the data corresponding to the hair of the person is removed because the sensor generates extremely noisy measurements there. Only a partial view of the face is available because of self-occlusions. For example, the left part of the nose is not visible. In the first experiment, we consider as features only the boundary of the face. Figure 12.c shows the segments that surround the face. Using this minimum information, the deformation of the surface initialized as a sphere (Figure 12.e) gives the shape shown in Figure 13.a. While the overall shape of the face was found, important facial features such as nose, mouth and eyebrow were smoothed so that only the general shape is recovered.

To avoid this smoothing effect it is necessary to tag the eye, nose and mouth as being important features. The thresholding of the magnitude of an edge detector on the intensity image provides a way to extract those features (Figure 12.b). A simple thinning process generates a new set of segment features that are used for a second experiment (Figure 12.d). The result of the final recovery is shown in Figure 13.b: this time the nose, eyebrow and mouth are clearly visible and the left part of the nose was interpolated. Several conclusions may be drawn from this experiment. First, it shows that fine details as well as general shape can be recovered due to the combination of feature and data forces. Second, it shows that the input features need not be very clean. Since a simple thresholding/thinning operator was used, no attempt was made here to extract clean edge features. Third, the algorithm provides a way to compress the input data drastically while retaining the important information. By using approximately 250 points (500 nodes for the whole surface) we are able to construct a fine and accurate description of the face.

Stacked Pebbles

The generality of our approach makes it particularly well-suited for unstructured environment in which there are few constraints on object shapes. The scene we use in this example consist in a set of stone pebbles that are lying on top of sand. In order to have more reliable data and

features, we took three range images from three different viewpoints. The features are of three natures: distance discontinuities, surface orientation discontinuities and shadow boundaries. Shadows are parts of the scene that are visible from the camera but are not illuminated by the projector. Shadows are used here as clues for the presence of an object since a shadow is created by an object occluding the light coming from the projector. The feature segments are shown in Figure 14.b. The figure demonstrates that using classical feature-based segmentation techniques would be a very challenging task.

The segmentation proceeds by selecting a shadow region, starting with the largest in the image. Using the geometry of the sensors as computed from an off-line calibration procedure, a 3-D point that is assumed to be inside the object is selected based on the position of the shadow in the image. The center of the initial sphere is initialized at this point, its radius is initialized to a constant value that is the average size of the expected objects. Figure 14.c shows the initial sphere. Data points and features that are less than a fixed distance away from the starting point are taken into account in the computation of the forces. The distance is computed from the radius of the largest object expected in the scene. Figure 14.d,e show the shape as it evolves. Figure 14.f shows the final shape.

This experiment validates several of our claims. First, shape is recovered even though there are few apriori constraints on object shape. Second, weak feature extraction is sufficient to recover shape. Most importantly, no scene segmentation is required. All that is needed is to hypothesize one point inside each object based on the knowledge of shadow geometry for this particular sensor. No feature-based segmentation is necessary. The algorithm is able to deal with numerous features and data points that are not part of the object. Finally, by using multiple views, this experiment shows that the algorithm works directly in 3-D and is independent of the image structure.

This application to the segmentation of natural scenes is part of an operational system that automatically picks up pebbles in cluttered natural environment[9]. The discrete surface extracted from range images are fed to a program that computes the stable grasp positions for a three-finger gripper. A manipulator executes the grasping. Figure 15 shows the gripper picking up a pebble in a typical environment. This shows the relevance of the algorithm in applications in which accurate surface descriptions of arbitrary objects are needed. In particular, the grasp determination algorithm uses the description of the surface to compute local properties such as curvatures. This would not be possible using standard parametric description such as superquadrics.

Stacked Cubes

To investigate further the ability of the algorithm to perform segmentation and surface reconstruction, we designed a simple scene with three overlapping cubes. Using simple objects of known shapes such as cubes allows us to compare the resulting segmentation and reconstruction with the actual scene more easily. Figure 16.a,b shows two opposite views of a scene where one cube is in balance between two adjacent cubes. The shape recovery was applied to each of the three cubes. The square drawn in the Figure 16.a, b, corresponds to the region that is used for the recovery of the first cube in both images. All features and data points that fall within this region of influence are taken into account in the recovery process. Therefore, the objects are recovered without explicit segmentation of the images. As before, the surface is initialized as a sphere of fixed radius whose center is roughly at the center of a group of features. The center of the sphere is also the center of the region of influence. Figure 16.c to Figure 18.a show features, range data,

initial surface position, and a shaded display of the recovered surface for each of the three objects. Figure 18.b shows the final description of the scene displayed as three shaded surfaces. This experiment shows that the scene was correctly segmented into objects with the correct shapes even though no explicit segmentation was done except for the selection of region of influence. Among the three reconstructed shapes, the second seems to be the best (Figure 17.d), mostly because he is not occluded by any other cube. On the contrary, the third cube (Figure 18.a) has one face that is poorly recovered. A look at the range data shows that there is no feature and no data corresponding to this face of the cube, most probably because of the occlusion from other cubes. In this case, the algorithm does the best job it can using only the smoothness energy. The reconstruction of the scene clearly demonstrates that all visible parts were correctly extracted. By computing for every node the minimum distance with range data, it is possible to identify which point was interpolated or not, and therefore to use only the reliable parts of the reconstructed surface later on.

4 Conclusion

References

1 M. BERTER, T. POGGIO and V. TORRE, Ill-Posed Problems in Early Vision, *M.I.T Art. Int. Memo 924*, 1987

2 P. BESL and R. JAIN, Segmentation Through Symbolic Surface Descriptions, *Proc. CVPR Miami*, pp 77-85, 1986.

3 R.BROOOKS, Symbolic Reasoning Among 3D Models and 2D Images. *Artificial Intelligence Journal*, 17:285-348, 1981.

4 C. CERRADA, K. IKEUCHI, L. WEISS, R. REDDY, A 3D Object Reconstruction System Integrating Range Image Processing and Rapid Manufacturing, Submitted to IEEE Robotics and Automation, 1990.

5 T. CHOI, H. DELINGETTE, M. HEBERT and K. IKEUCHI, A Perception and Manipulation System for Collecting Rock Samples, SOAR,1990.

6 H. DELINGETTE, M. HEBERT and K. IKEUCHI, A Perception and Manipulation System for Manipulation in Natural Environments, Submitted to IEEE Robotics and Automation, 1990.

7 F. FERRIE, J. LAGARDE and P. WHAITE, Darboux Frames, Snakes and Superquadrics: Geometry from the Bottom-Up, IEEE, 1989.

8 C. FOX, An Introduction to the Calculus of Variations, *Dover Publications Inc.*, 1963.

9 C. FRANCOIS, K. IKEUCHI and M. HEBERT, A Three-Finger Gripper for Manipulation in Unstructured Environments, 1990.

10 A. GUPTA, L. BOGONI and R. BAJCSY, Quantitative and Qualitative Measures for the Evaluation of the Superquadrics Models, IEEE, 1989.

11 K. IKEUCHI and B. HORN, Numerical Shape from Shading and Occluding Boundaries, *Art. Int. 17*, pp141-184, 1981.

12 M. KAAS, A. WITKIN and D. TERZOPOULOS, Snakes: Active Contour Models, *Intern. Journ. of Computer Vision*, pp321-331,1988.

13 D. MARR and K. NISHIHARA, Representation and Recognition of the Spatial Organization of Three Dimensional Shapes, *Proc. Roy. Soc. London*, 200:pp269-294,1978.

14 A. PENTLAND, Recognition by Parts, *Proc. ICCV London*, pp 612-620, 1987.

15 A. PENTLAND, Perceptual Organization and the Representation of Natural Form, *SRI International Tech. Rep. no 357*, 1986.

16 A. PENTLAND and S. SCLAROFF, From Features to Solids, *M.I.T Media Lab Tech. Rep. No 135*, 1990.

17 K. SATO and S. INOKUCHI, Range-Imaging System Utilizing Nematic Liquid Crystal Mask, IEEE, pp 657-661.

18 D. TERZOPOULOS, A. WITKIN and M. KAAS, Symmetry-Seeking Models for 3D Object Reconstruction, IEEE, 1987.

19 D. TERZOPOULOS, Computing Visible-Surface Representations, *M.I.T Art. Int. Memo 800*, 1985.

20 A. WITKIN, M. KAAS, D. TERZOPOULOS, Physically Based Modeling for Vision and Graphics, *Int. Journ. of Computer Vision*, pp321-331, 1988.

21 N.YOKOYA, M. KANETA and K. YAMAMOTO, Recovery of Superquadrics Primitives from Range images by Simulated Annealing, *ETL Tech. Rep. TR-90-18,* 1990.

22 S. ZUCKER, The Organization of Curve Detection: Coarse Tangent Fields and Fine Spline Coverings, IEEE 1988.

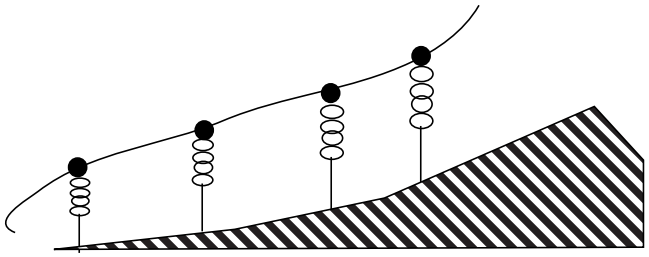


Figure 1 . Nodes of a spline are attached to data points via springs

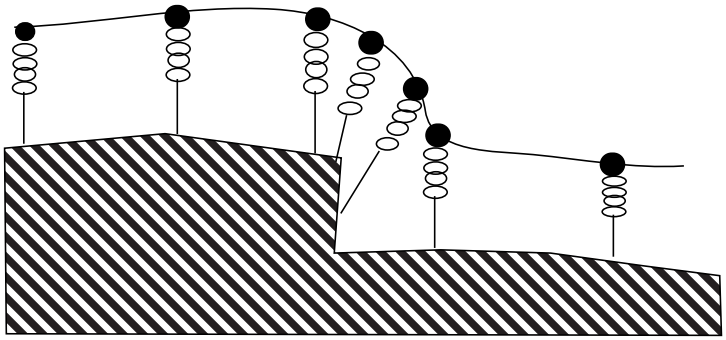


Figure 2 Optimum mapping: the nodes are concentrated near the feature, here an edge

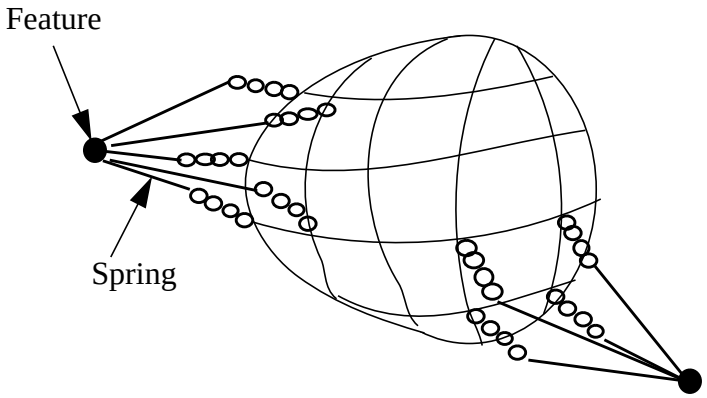
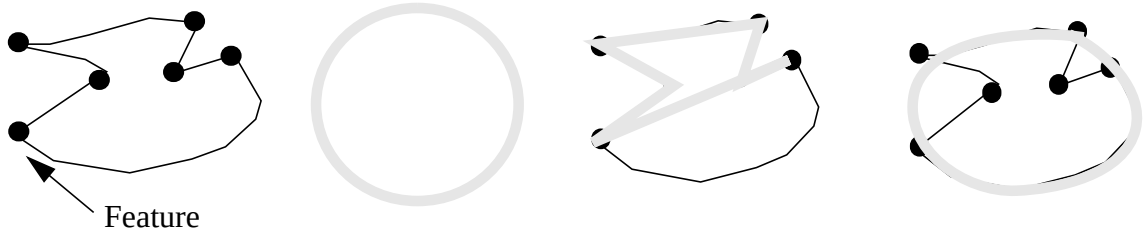


Figure 3 For every feature, we attach springs to all nodes of the surface



(a) (b) (c) (d)

Figure 4 **a,** An object with its features (normal discontinuities); **b,** Surface initialization; **c,** Final shape if feature deformation is predominant; **d,** Final shape if data deformation is predominant

(a) (b)

Figure 5 **a,** Curve U(x); **b,** Evolution of $E_{\text{feature i}}$ as a function of time and distance between surface point and feature

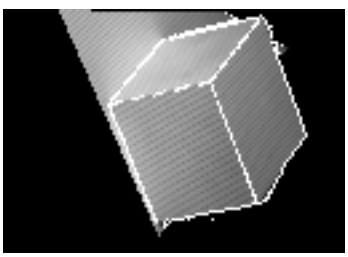
(a) (b)

(c) (d)

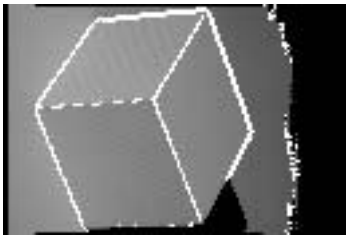
Figure 6 **a,** Features and initial position of the surface; **b**, Initial potential field; **c,** Intermediate potential field; **d,** Final potential field;

Figure 7 Curve W(x)

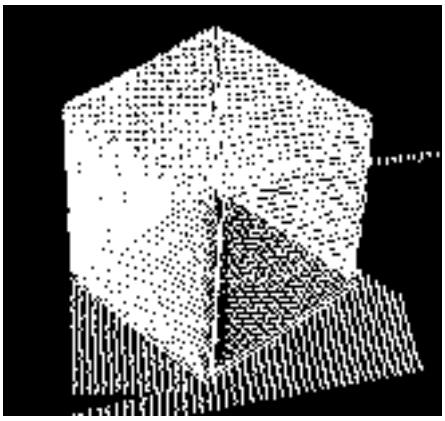
Figure 8 Tessellated icosahedron



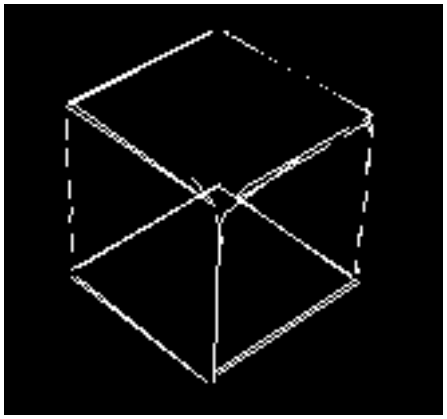
(a)



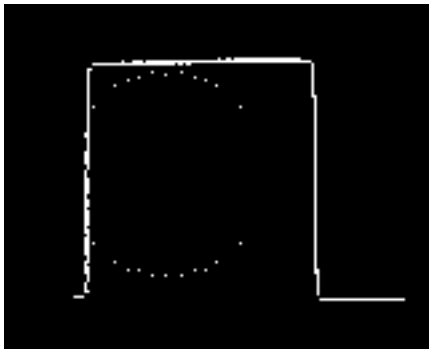
(b)



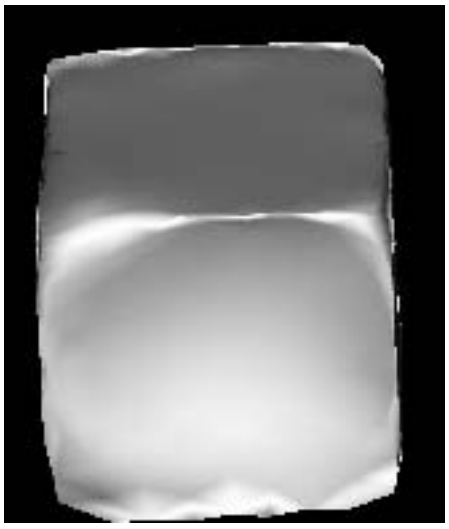
(c)



(d)



(e)



(f)

Figure 9 **a**, Intensity image with features from the first view; **b,** Intensity image with features for the second view; **c,**Range data from thetwo views combined; **d,** Features from the two views combined; **e,** Cut-section of the initial position of the surface; **f,** Final shape of the surface;

Figure 10 Final triangulated surface

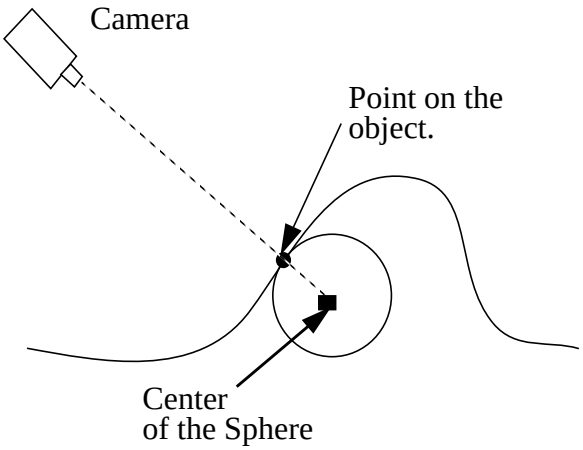
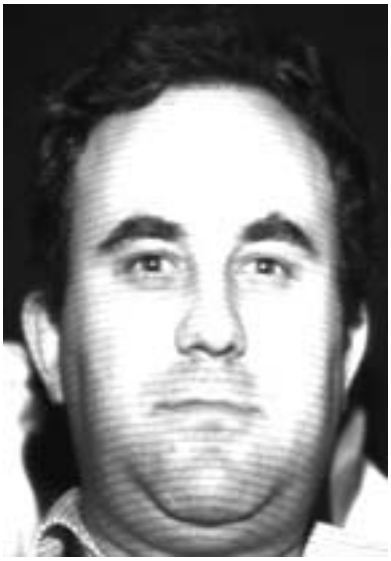
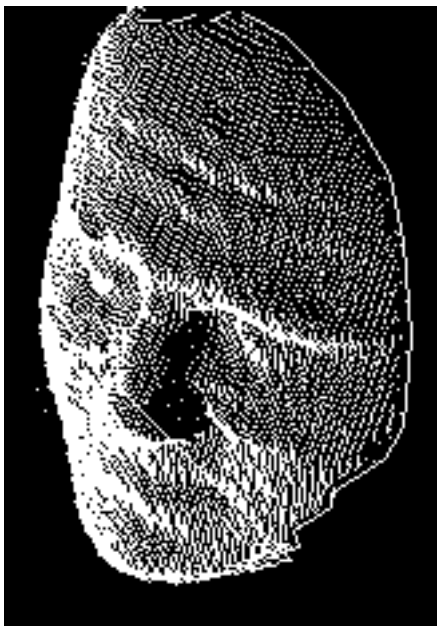


Figure 11 Initialization of the sphere



(a)

(b)

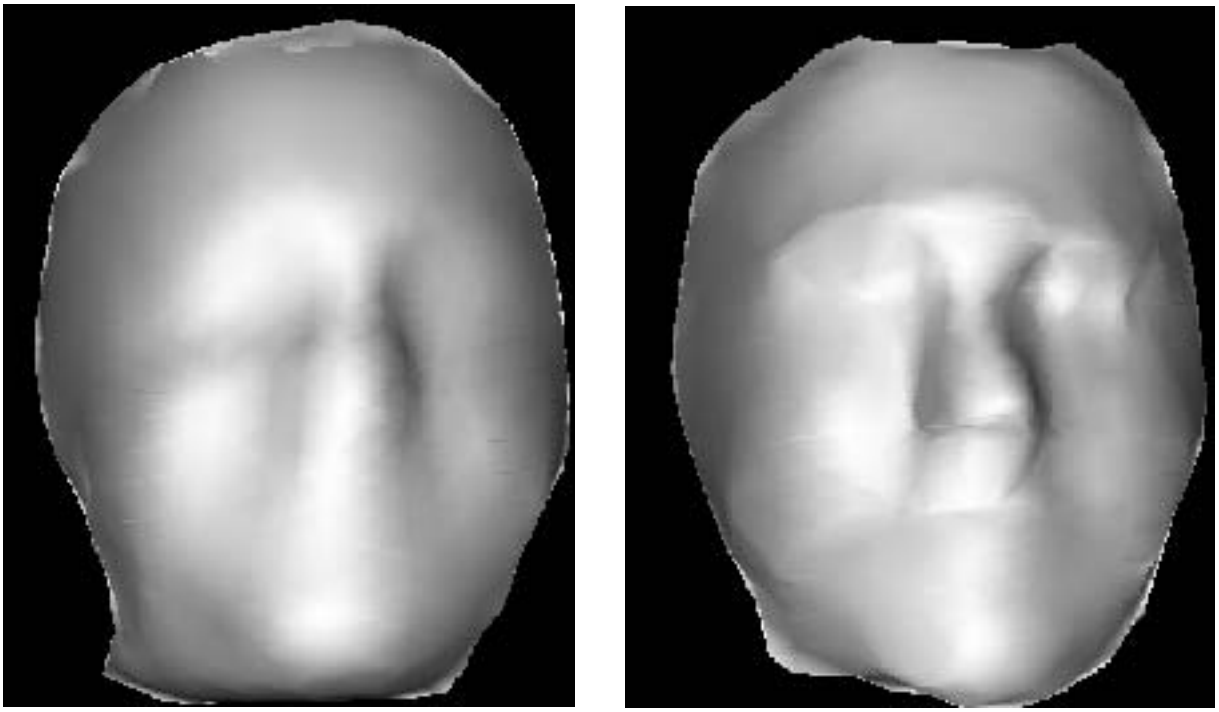


(c)

(d)

(e)

Figure 12 **a,** Intensity image; **b,** Features extracted from intensity image; **c,** Range data with features; **d,** Squelletization of the features; **e;** Initial position of the sphere;



(a) (b)

(a) (b)

Figure 13 **a,** Reconstruction of the face using only contour features; **b,** Reconstruction of the face using intensity features;

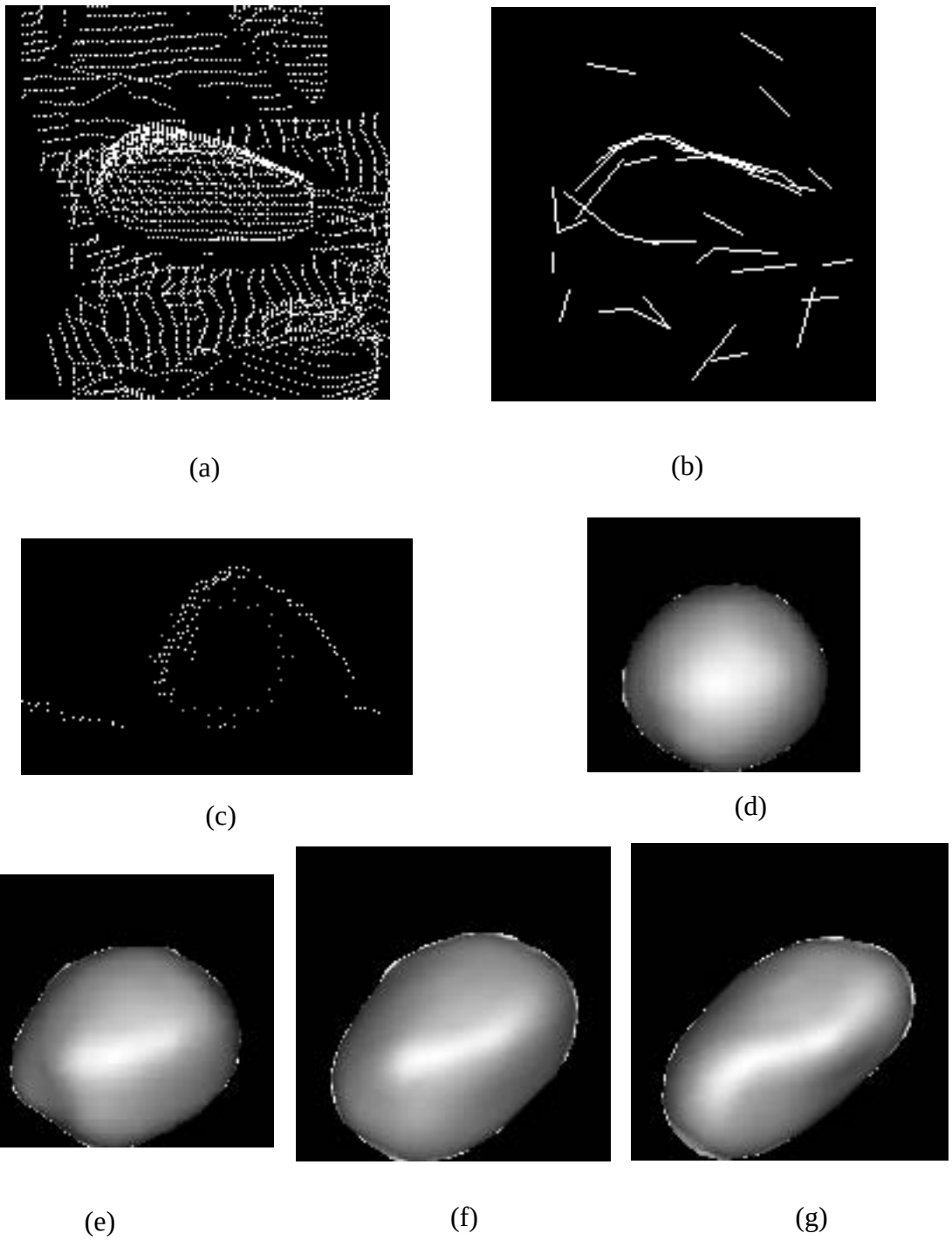


Figure 14 **a,** Range data coming from three different views; **b,** Feature segments; **c,** Cu-section of the initial position; **d,** Surface after 15 iterations; **e,** Surface after 45 iterations; **f,** Surface after 75 iterations; **g**, Final shape after 150 iterations;

Figure 15 Using surface models for manipulation

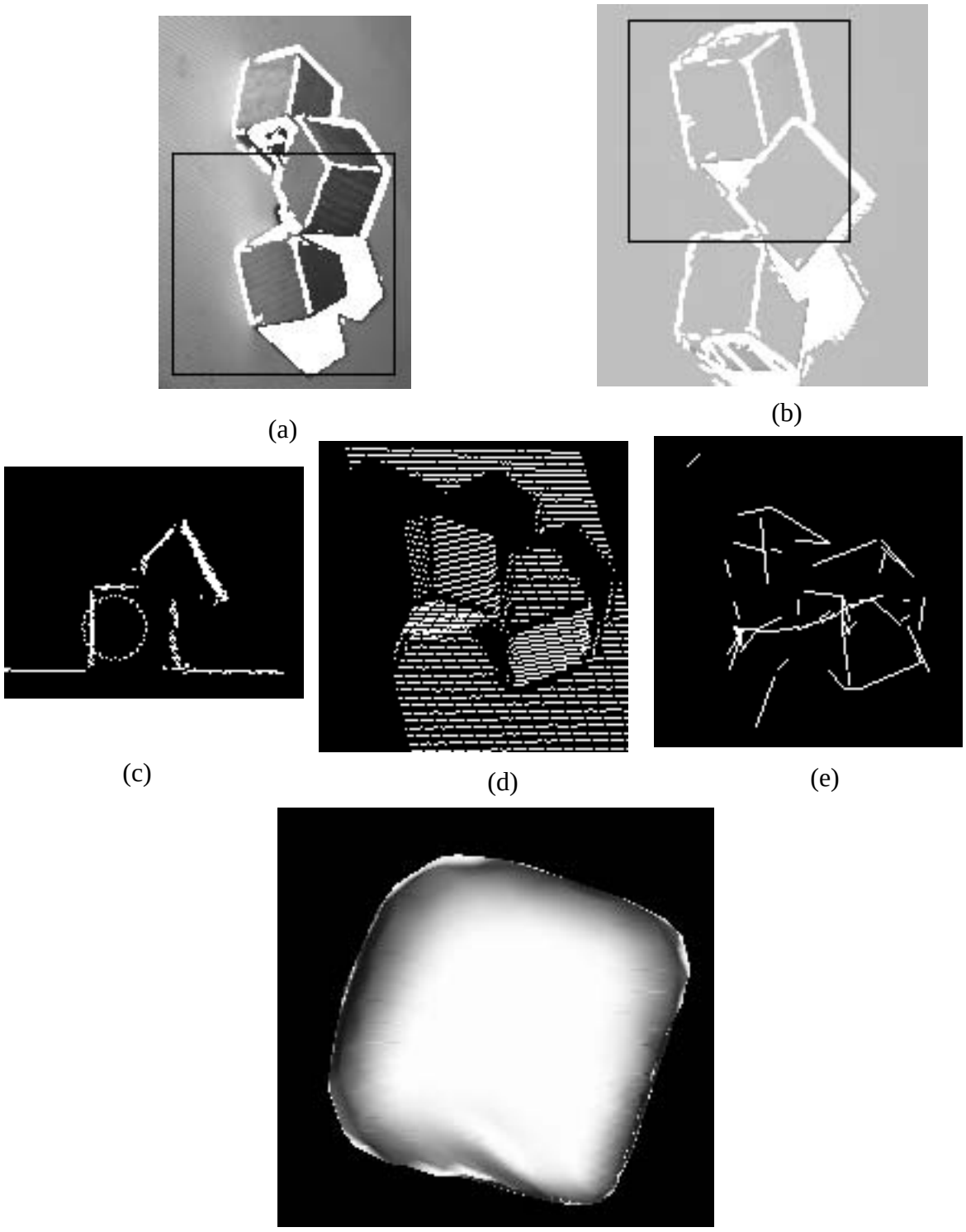
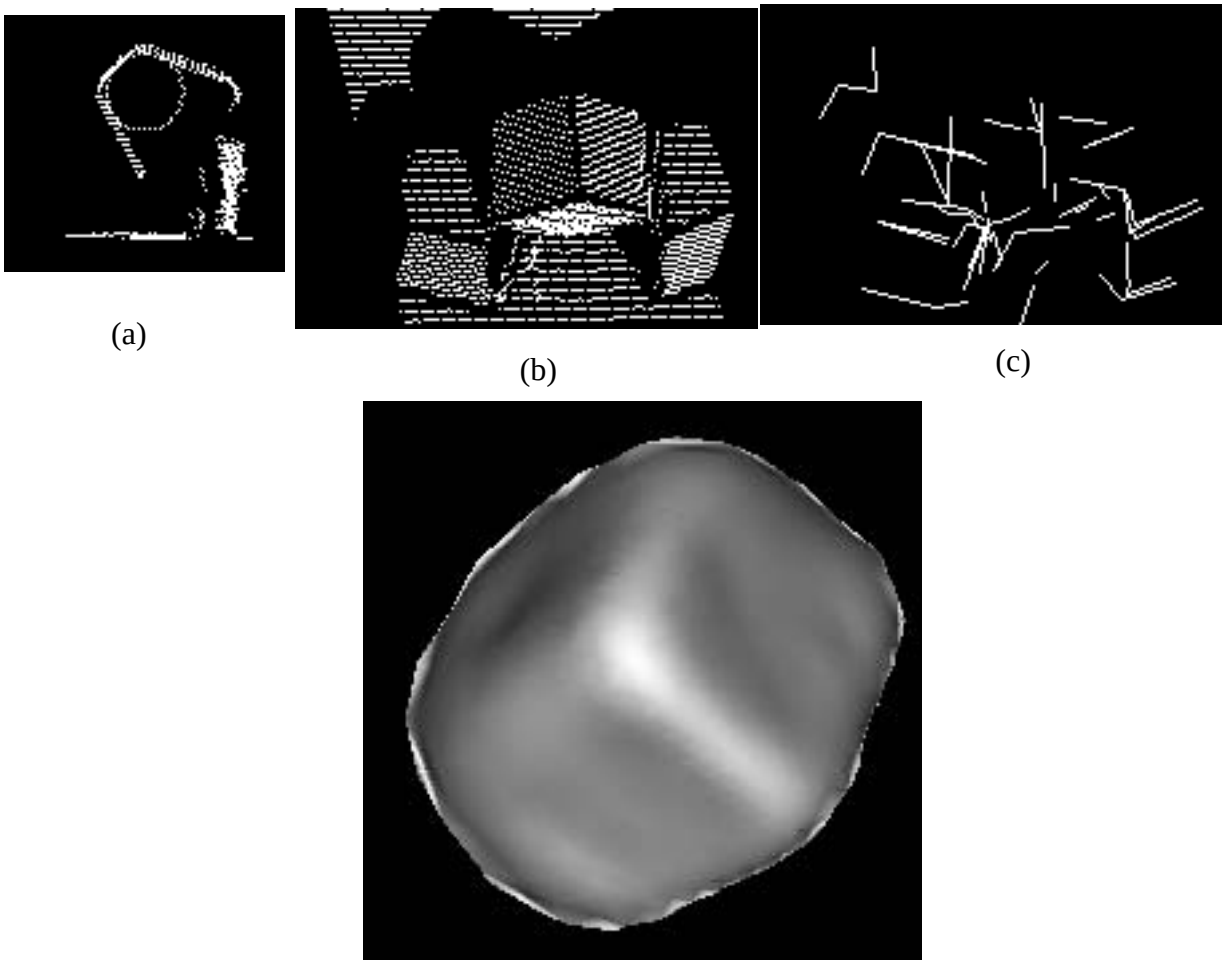


Figure 16 **a,** First range image with features; **b,** Second range image with features; **c, d,** and **e,** Initialization, range data and features for the extraction of the first cube; **f,** First cube reconstructed by the deformable surface;

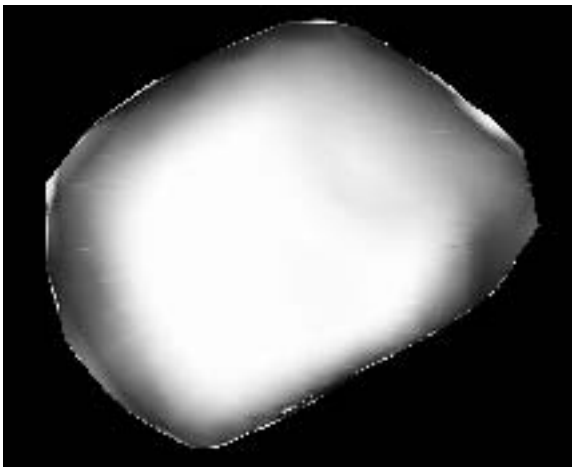


(a) (b) (c)

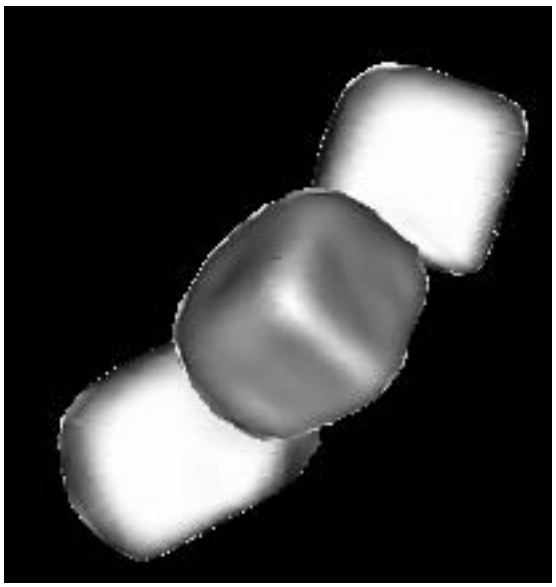
(d)

(e) (f) (g)

Figure 17 **a, b,** and **c,** Initialization, range data and features used for the extraction of the second cube; **d,** Second reconstructed by deformable surface; **e, f, g,** Initialization, range data and features used for the extraction of the third cube;



(a)



(b)

Figure 18 **a,** Third cube reconstructed by deformable surface; **b,** Reconstruction of the scene;