Toward 3D Vision from Range Images: An Optimization Framework and Parallel Networks

Stan Z. Li

Vision, Speech and Signal Processing Group
Department of Electronic and Electrical Engineering
University of Surrey
Guilford, Surrey GU2 5XH, U.K.

S.Z.Li@ee.surrey.ac.uk

Abstract

We propose a unified approach to solve low, intermediate and high level computer vision problems for 3D object recognition from range images. All three levels of computation are cast in an optimization framework and can be implemented on neural network style architecture. In the low level computation, the tasks are to estimate curvature images from the input range data. Subsequent processing at the intermediate level is concerned with segmenting these curvature images into coherent curvature sign maps. In the high level, image features are matched against model features based on an object description called *attributed relational graph* (ARG). We show that the above computational tasks at each of the three different levels can all be formulated as optimizing a two-term energy function. The first term encodes unary constraints while the second term binary ones. These energy functions are minimized using parallel and distributed relaxation-based algorithms which are well suited for neural network implementation. Some experimental results are presented for the curvature-based segmentation, ARG matching and 3D surface matching.

List of Symbols

$a^k(i,j)$	The k^{th} relation between nodes i and j of an ARG
α	The threshold for detecting depth discontinuities
C	Charging capacitance of a neural network
$\mathbf{D}$	The domain of a function
$\mathbf{D}_I$	A set of labels, at intermediate and high levels
d	Input data to an optimization engine
$d(i)$	Datum i, at low level
$d(i,I)$	Datum (i,I), at intermediate and high levels
E	A cost energy function(al)
E^c	The closeness term of E
E^s	The smoothness term of E
E_Λ	Energy function for matching the scene and model Λ
f	The solution function of an optimization problem
$f(i)$	Reconstructed surface point i, at low level
$f(i,I)$	Plausibility that i is labeled I, at intermediate and high levels
$f^{(0)}$	Initial state of f
f^*	The final solution, $f^* = f^{(\infty)}$
$g(\cdot)$	A function relating to the definition connection π
$\theta(u)$	The input-output sigmoid function in the Hopfield model
$\mathbf{g}$	An attributed relational graph (ARG)
H	Mean curvature
HK	$H-K$ curvature label
K	Gaussian curvature
λ	The weight between E^c and E^s in E
μ	A constant in a dynamic equation
#	Cardinal number of a set
$NULL$	A special node, *i.e.* node 0, in a model ARG
ω	A neighborhood system
$\omega(i)$	The set of neighbors of i, at low level
$\omega(i,I)$	The set of neighbors of (i,I), at intermediate and high levels
π	The connection of a network
$\pi(i,j)$	The connection strength between i and j, at low level
$\pi(i,j;u,v)$	The connection strength between (i,j) and (u,v), at low level
$\pi(i,I;j,J)$	The connection strength between (i,I) and (j,J), at intermediate and high levels
Ψ	An updating operation in relaxation labeling
$q(i)$	A component of energy gradient $\frac{1}{2}\nabla E$ at i
$q(i,I)$	A component of energy gradient $\frac{1}{2}\nabla E$ at (i,I)
R	Resistance of a neural network

$\mathbf{R}$	The set of real
r	An integer concerning the size of neighborhood
$\rho(i,j)$	A distance between i and j
$\mathbf{S}$	A solution space
$\mathbf{S}^+$	An augmented solution space
σ^k	Parameter in computing connection π from the k^{th} relation
$T(i,j)$	Conductance between neurons i and j
τ	A threshold for a curvature sign map from curvature image
$v^k(i)$	The k^{th} attribute of node i of an ARG
$z(x,y)$	A graph surface point at (x,y)

Contents

1 Introduction

Three-dimensional computer vision aims at understanding the 3D world by analyzing input images. Marr [67] believed that *depth* information should be made explicit to achieve higher vision goals. Most computer vision research has been based on this belief. However, unlike human vision which is able to infer depth quickly and easily from intensity images, achieving 3D computer vision goals from intensity images presents formidable difficulties due to lack of constraints for solving visual computation problems therein. As the sensor technology advances, range images from commercial range scanners [53, 74] have been available. A range image or depth map is an array of pixel values which represent the distances from the sensed focus plane to object surfaces within the field of view. It provides explicit range information about visible surfaces on the image grid, and is largely independent of the illumination conditions of the environment and the reflectivity of the objects being sensed. Therefore, many perceptual problems arising from intensity images do not occur from range images. Because of these advantages, the last few years have seen increasing interest in 3D vision from range images. Despite the advantages, however, range image understanding still remains difficult and presents theoretic and pragmatic challenges [17, 51].

The first problem is how to represent 3D objects. This is a fundamental and yet unsolved issue not only in computer vision but in cognitive sciences in general. A second problem is how to extract the representation from images and subsequently match it against potential models. This has motivated most computer vision research. A final problem is the actual implementation of the computational algorithms. These three problems are associated with the three levels of the computational theory stated by Marr [67]. The work reported here attempts to study these problems in 3D vision from range images.

In this work, we use a surface-based representation. This is because surface shapes are important in understanding images, as recognized by the popularity of research on shape from X [47]. Explicit visible surface information is the advantage we should take from range data. We desire a representation scheme which is invariant to viewpoint to facilitate object matching, recognition and other computer vision tasks such as pose determination. Furthermore, we choose to use a shape representation based on surface curvature properties, [3, 6, 18], such as the sign (-,0,+) of mean curvature and Gaussian curvature. These properties are invariant to rotations and translations. We do not consider scale-invariance to be a problem since the absolute scale is usually to hand in range data. In this representation, objects are decomposed into surface patches segmented based on curvature signs, and are described by attributes of, and relations between, these patches. This results in a graph-like description which we shall term an *attribute relational graph* (ARG).

There are three primary difficulties in extracting the curvature-based representation used in this work. The first is that curvature is computed from derivatives of up to second order, which is highly noise-sensitive as widely reported [7, 29, 43, 97]. The second difficulty is the oversmoothing effect introduced by the required noise suppression operations; it is this smoothing across discontinuities that distorts the underlying surface shapes. The third problem is that of attempting to organize initially segmented coarse curvature sign maps, which are usually fragmentary, into a form consisting of coherent surface patches.

The matching between image features and models at the high level is done based on ARGs extracted from the input image and models. A problem in relational matching is that it often suffers from errors and mistakes in extracted relational descriptions. Even a very small number of missing or false relational measurements may incur the total failure of the whole matching process. This is of a general problem in computer vision and is not specific to the application described here.

To solve these problems, we develop a three level system. At the low level, surface curvature images are estimated; at the intermediate level, the estimated curvature images are segmented into curvature sign maps; and at the high level, the input image is matched to models based on the image ARG derived from segmented surface patches and model ARGs built from known model surfaces. To prevent surface shape distortion due to oversmoothing at depth edges, we pay particular attention to the role of discontinuities during the image curvature computation at the first-level and also during its subsequent segmentation at the second-level. To prevent possible failure of matching at the high level due to the violation of some "hard" or symbolic relational constraints such as adjacency, we choose instead to use "soft" or numerical constraints, such as the minimum distance between patches, and formulate an algorithm for matching these constraints.

We propose a unified optimization framework for solving the problems at all the three levels of interpretation. Within this framework, the problem solution is found by minimizing an energy function. The energy functions for all three levels of processing have the same form. They consist of two terms, *i.e.* a "closeness" term and a "smoothness" term. The former encodes *unary* constraints while the latter encodes *binary* constraints [60]. The formulation is related to continuity-controlled regularization [13, 38, 58, 69, 89], relaxation labeling [26, 49, 57, 83, 92], and inexact subgraph matching [10, 22, 28, 32, 96]. The minimization of these energies is performed by using relaxation algorithms which are inherently parallel and distributed. These algorithms are well suited for neural network style implementation.

The outline of this paper is as follows. Sections 3 and 4 constitute the computational framework. In Section 3, we formulate the processing at each of the three levels as an energy function minimization task. In Section 4 we develop algorithms for minimizing the energy functions. In Section 5, we discuss potential neural implementation of the algorithms. However, before proceeding to these topics we will give an introductory illustration on the representation that we use together with an overview of our system structure (section 2). Our aim in doing this is to bring out computational issues involved in the process. We will also show some experimental results on the curvature-based segmentation and the ARG matching, during describing the representation in the next section and the main ideas of our own work in Sections 3, 4 and 5 to illustrate the representation and the ideas. In Section sec:exp, we will present two other experiments to demonstrate the processing from a range image to 3D surface matching. After that, we will illustrate our choice of parameters involved in all experiments we have presented and the times spent for the computation. Having presented the main contents of this work, we will review the related literature in Section 7. Finally in Section 8, we will emphasize the novelty of our contribution.

2 Representation and System Structure

2.1 Surface Curvature

According to the fundamental *existence and uniqueness theorem* of surfaces from *differential geometry* [65], the local shape of an arbitrary smooth surface is uniquely determined by the six parameters in the first and the second fundamental forms of the surface. The information contained in these six parameters can be properly expressed in terms of the mean curvature and Gaussian curvature. The mean curvature and the Gaussian curvature are invariant to rotations, translations and surface parameterization as long as the Jacobian of the transformation is non-zero. Given a graph surface $\{(x, y, z(x, y))\}$, the surface curvature can be computed from five partial derivatives of z, *i.e.* z_x, z_y, z_{xx}, z_{yy}, z_{xy}

$$H = \frac{(1 + z_y^2)z_{xx} + (1 + z_x^2)z_{yy} - 2z_x z_y z_{xy}}{2(1 + z_x^2 + z_y^2\ )^{3/2}} \tag{1}$$

$$K = \frac{z_{xx}z_{yy} - z_{xy}^2}{(1 + z_x^2 + z_y^2)^2} \tag{2}$$

where H and K represents the mean curvature and the Gaussian curvature, respectively.

2.2 Surface Segmentation Based on Curvature

Some form of symbolic descriptions have to be derived in order for high level mechanisms to use these curvature properties. One possibility is to use the *sign* of both curvatures. It is invariant to rotation, translation *and* scale-changes. The combinations of the sign of mean curvature and the sign of Gaussian curvature yield eight possible surface types [6] as shown in Table 1.

Table 1: ===== about here =====

Segmenting a range image into curvature sign maps generally consists of two main stages: computing the surface curvature from range data; and segmenting the resulting images into curvature sign maps. Great difficulties have been reported in accomplishing these tasks. The first stage involves estimation of derivatives of up to the second order, which is mathematically ill-posed [80], and is highly noise-sensitive as widely reported [7, 29, 43, 97]. The second stage is necessary to recover from noise-induced errors made during the first stage of processing; it organizes degraded symbolic maps into "coherent" ones consisting of meaningful surface patches. The lowest level of our system aims to perform the estimation of curvature images while the intermediate level performs the segmentation of the estimated curvature images.

Figure 1: ===== about here =====

Figures 1 and 2 show two results of surface segmentation based on the curvature signs produced by the the first two levels, contained in the latter image being two superimposed objects. Shown at the top of each figure are the original range image (the image is shaded for easier visual interpretation), regularized image and histograms of the estimated curvature images. In the middle row are the coarse curvature sign maps obtained by thresholding the estimated curvature images from the low level; they are, from the left to the right, the coarse mean curvature (H) sign map, the coarse Gaussian curvature (K) sign map and the coarse HK combination map. The first two maps are the input to the intermediate level. At the bottom are the refined maps by the intermediate level. Significant improvement on the coherence of the segmentation can be seen in the transition from the coarse maps to the refined maps. For more segmentation results from images of various complexity, readers are referred to [62].

The refined HK map identifies up to eight different surface types which are independent of rotations, translations and scale changes. From this, an attributed adjacency graph (ARG) can be built to give an invariant description of the scene in the image. Model objects are also represented in the same way. The higher level processing then takes input computed from the scene ARG and each of the model ARGs and performs surface matching and object recognition.

Figure 2: ===== about here =====

2.3 Attributed Relational Graphs

An attributed relational graph (ARG) is a graph-like description of a scene or a model. It is denoted by $\mathbf{g} = (v, a)$ in which $v = \{v(i) \mid 1 \leq i \leq m\}$ is a set of m attributed nodes, $a = \{a(i,j) \mid 1 \leq i,j \leq m\}$ is a set of relations between the nodes. For node i, $v(i) = \{v^k(i) \mid 1 \leq k \leq K1\}$ represents node (surface patch) attribute *types* $k = 1, \ldots, K1$ of patch i; it also represents corresponding attribute *value* $v^k(i)$. An attribute is a unary property which can be either symbolic or numerical. The attributes we have used are

- $v^1(i) = hk(i)$: HK label of patch i
- $v^2(i) = area(i)$: visible 3D surface area of patch i

For a pair of nodes i and j, $a(i,j) = \{a^k(i,j) \mid 1 \leq k \leq K2\}$ represents binary relations $k = 1, ..., K2$ between patches i and j with corresponding relational value $a^k(i,j)$. A relation can be symbolic or numerical. The relation we have used is

- $a^1(i,j) = dist(i,j)$: the minimum (nearest) 3D Euclidean distance between patches i and j

which takes a real value. The reason why we choose the minimum (nearest) distance instead of symbolic adjacency is that the former is more robust than the latter under occlusion and noise. Although the latter could provide more concrete constraints for matching, it can often be violated because of occlusion and/or segmentation error due to noise. For example, one patch may be split into many; additional patches which are not in the model may appear; and two connected patches may appear disconnected in some views. We use "soft" constraints to prevent possible matching failures due to unsatisfied "hard" constraints.

Figure 3: ===== about here =====

Figure 4: ===== about here =====

There are a number of criteria for the representation of shapes [18, 68]. Our choice of attributes and relations is based mainly on the requirement of accessibility, invariance, robustness and locality. Other less local information can also be included in ARG. For example, Fan *et al.* [25] used the following attributes

- $v^3(i)$: average mean curvature over patch i
- $v^4(i)$: average Gaussian curvature over patch i

and relations

- $a^2(i,j)$=the 3D distance between centroid of patches i and j
- $a^3(i,j)$=the angle between orientations of patches i and j

where the orientation of a patch is defined as the direction of the normal for planar patches, the direction of the axis for cylindric and conic patches, or the normal at the curvature extremum otherwise. The angle between surface normals of planar patches are important for matching planar patches.

Although our representation is based on surfaces, information on boundaries, feature curves and volumes can provide important supplementary constraints to our surface-based scheme. The result is a more complete object description which includes zero-crossings of the Gaussian curvature, zero-crossings of the maximum principal curvature, maxima of the maximum principal curvature [71], lines of curvature, asymptotes, surface intersections [19], *etc.* These curves lie between nearby patches and can be used to characterize types of relations between them.

Object recognition schemes proposed to date fall into three categories [40]: (1) those based on invariant properties, (2) those based on object decomposition into parts and (3) those based on the process of explicitly compensating for the transformation between the viewed object and the stored model. Our work falls into the first category. Representations by parts, *e.g.* in [42, 76, 86, 95], provide an easier way to describe object hierarchies. However, recognition by parts has to be preceded by the recognition of parts and there is a big gap to be filled *from images to parts.*

In this work, several sources of information are used to build a scene ARG. Firstly the refined HK map from the intermediate level is used to extract the HK label and the surface area of each patch and the relations between patches. Secondly a discontinuity map π from the lowest level is used to provide additional information on the segmentation. Thirdly, the original range map d itself and first derivative maps f_x^* and f_y^* is used to extract 3D surface area. Figure 3 shows two cup images and their HK maps. Note that the image at the bottom is noisier and therefore its HK map is fragmented. Figure 4 are the ARGs built from the cup HK maps. For clarity, only relations with zero distance, *i.e.* those which are directly adjacent to each other, are displayed. However, there is a weighted link between every pair of nodes.

Model ARGs are pre-stored, *i.e.* built off-line. They can be built by using CAD modeling techniques [16, 9, 30, 35, 17], or integrating multiple views [93, 95]. We have yet been able to build complete 3D object model ARGs. However, we have created scenes for experiments on the curvature-based segmentation and 3D surface matching under rotation and occlusion (Section 6).

The high level aims at matching the ARG from the image to the ARG of a model. The matching is based on the unary attributes, *i.e.* HK label and 3D area, and the binary relation, *i.e.* the minimum distance. The HK sign label are invariant to rotations, translations and scale changes. The 3D surface area of a curvature-sign labeled patch in the scene generally should not exceed its counterpart in the model, being smaller if occlusion occurs. The minimum distances between patches provide a mutual constraint on their relative location. This constraint is invariant to rotations and translations but not to scale changes and does not change dramatically because of occlusion. An ARG constructed in this way is to a certain extent an invariant structural representation of an object. For this reason, the problem of matching between model surfaces and scene surfaces has been reduced to that of inexact relational graph matching. If we take Figures 4a and b as the model object and the scene object, respectively, then the correct matching from the scene to the model, *i.e.* from b to a, should be $5 \rightarrow 1$, $1 \rightarrow 2$ and the rest to $NULL$, the $NULL$ being a special model node representing unknowns.

In the rest of this paper, when referring to the same concepts pertaining to both data and models, we will use lower-case notation to refers to the image or scene, *e.g.* $m, hk(i), a(i,j)$ and upper-case notations to refer to models, *e.g.* $M, HK(I), A(I,J)$, *etc.*

Figure 5: ===== about here =====

2.4 Overview of System Structure

Our system consists of three levels: the low level, the intermediate level and the high level, as illustrated in Figure 5. In order to progress from one level to another, some processing operations are required. Between the low and the intermediate levels, there is a thresholding operation on curvature images. Between the intermediate and the high levels, there is a procedure that constructs an ARG from HK label map and other maps. The function of each block is described as follows.

- The low level estimates mean curvature (H) and Gaussian curvature (K) from the input range image (at location (1) in Figure 5), resulting in curvature images (at location (2)). The core of this level is the discontinuity-controlled surface reconstruction process.

- The curvature images are thresholded into coarse curvature sign maps (at location (3)).
- The intermediate level refines the coarse curvature sign maps obtained from the automatic thresholding to increase the coherence of curvature sign patches, resulting in a refined HK sign combination map (at location (4)) which consists of refined surface patches of up to eight different surface types. The basic function of this level is to restore degraded multi-level images to form coherent regions.
- An attributed relational graph which symbolically describes the visible surfaces in the image is then built. This combines information from the refined HK map, the first derivative maps, the depth discontinuity map (at location (2')) and the range map.
- The high level matches the scene ARG (at location (5)) against each of object model ARGs, resulting in pairings between surface patches in the scene and those in each model (at location (6)). Subsequent object recognition can be done by selecting the best matched model. The core of this level is inexact sub-ARG matching.

The input to each of these three *computational levels* are, respectively, the range image, the coarse curvature sign maps and the ARGs. However, these are not the actual inputs to each of the corresponding *computational networks*, or *optimization engines*. There are further substructures in the system.

The computational process from range image to curvature images consists of two main sequential phases: the estimation of first-derivative images from the range image, and the estimation of second-derivative images from the estimated first-derivative images. Each phase is a regularization process aimed at estimating derivative images of its input. The curvature computation process is illustrated in Figure 6. Firstly, the range image is regularized. The first derivative estimates f_x, f_y are computed from the regularized range image f^* using the finite difference formulation

$$f_x(i,j) = f^*(i+1,j) - f^*(i,j)$$
$$f_y(i,j) = f^*(i,j+1) - f^*(i,j)$$

Next, the first derivative images are regularized to increase the reliability of computing the second derivative estimates f_{xx}, f_{yy} and f_{xy} using finite differencing. The subsequent computation of the mean curvature image H and the Gaussian curvature image K is straightforward; it is achieved by substituting f^* for z in Eq. (1) and (2) In Figure 6, the estimated images marked by an asterisk are to be used for the curvature computation.

Figure 6: ===== about here =====

The intermediate level refines its inputs (the coarse H map and K map) independently and parallelly. The output is the refined H, K and thus HK maps. The high level matches the scene ARG against each of the model ARGs. The actual input to the optimization engine of this level is computed from the scene ARG and a model ARG. These two ARG's also determine the inter-connections of the computational network. By contrast, the inter-connections of the networks at the first two levels are determined by the depth discontinuities of the input range image. The output of the high-level network is a list of matches, or mapping, from the scene ARG to the model ARG, which can be many-to-one.

3 Energy Functions

In this framework, the solutions of the problems are found by minimizing energy functions using relaxation algorithms. These energy functions consist of a weighted sum of two terms: one term encoding unary constraints and the other binary constraints. This formulation is closely related to the energy function used in regularization techniques [90] which have been widely used for solving ill-posed problem in early vision [4, 80]. It is also akin to relaxation labeling [26, 49, 57, 83, 92] where the criterion is usually composed of a term encoding binary constraint.

A regularization solution f^* from data d is found by minimizing an energy function

$$E(f^* \mid d) = \min_f E(f \mid d)$$

where E is a weighted sum of two terms

$$E(f \mid d) = E^c(f \mid d) + \lambda E^s(f \mid \omega) \tag{3}$$

The first is a *closeness* term that measures the closeness of the solution f to the input data d

$$E^c(f \mid d) = \sum_i [f(i) - d(i)]^2$$

The second is a *smoothness* term, or regularizer, that measures the smoothness of the solution f with respect to a neighborhood system ω

$$E^s(f \mid \omega) = \sum_i \sum_{j \in \omega(i)} [f(i) - f(j)]^2$$

where $\omega = \{\omega(i) \mid \forall i\}$ defines the neighborhood of i. The weight λ controls the relative importance of the two. The smoothness term is imposed as a constraint on the solution based on a *a priori* assumption that the physical world is in general smooth.

The smoothness assumption, however, is not always true throughout the world. It is violated, for example, at discontinuities. Uncontrolled regularization across discontinuities, depth discontinuities in particular, often leads to unrealistic results because of the smoothing spline nature of the regularizers [79]. This problem may be overcome by modifying Eq. (3) to include a discontinuity controller π into the smoothness term E^s. This results in

$$E(f \mid d, \pi, \omega) = E^c(f \mid d) \ + \ \lambda E^s(f \mid \pi, \omega) \tag{4}$$

where

$$E^s(f \mid \pi, \omega) = \sum_i \sum_{j \in \omega(i)} \pi(i,j)[f(i) - f(j)]^2$$

is a discontinuity controlled smoothness term [13, 69, 89] The controller $\pi = \{\pi(i,j) \in [0,1] \mid \forall i,j\}$ is a connection function which controls the degree of interaction between points i and j.

Eq. (4) is the basic form that our energy functions take at every of the three levels. The solution f^* to a computational problem in the process is found by minimizing such an energy function

$$E(f^* \mid d, \pi, \omega) = \min_f E(f \mid d, \pi, \omega) \tag{5}$$

In this section, we will, for each of the three levels, illustrate the meaning of the input d and the solution f, and formulate the connection π, the neighborhood system ω and the particular form of the two functional terms. This section, together with the next section on minimization algorithms, constitutes the computational framework.

As can be seen, conceptually there are close parallels among the three energy functions. For a convenient comparison of these parallels, we will use the same notation to represent the relevant concepts at all three levels. For instance, we will use f to denote a solution, d an input, π a connection matrix, ω a neighborhood system, *etc.* However, the notation is valid and precise only at the level presently discussed unless otherwise specified. For the sake of clarity, our discussion on the low and intermediate levels will focus on one-dimensional "images". Two-dimensional formulae are direct extensions of the one-dimensional versions.

3.1 Low level

At this level, the computation is done on a lattice domain $\mathbf{D} = \{i \mid 1 \le i \le m\}$ where m is the size of the image. The function $f = [f(i)] \in \mathbf{R}^{\mathbf{D}}$ is the solution to be found. The final f^* is a reconstructed depth image or an estimated derivative image (see Figure 6). The function $d = [d(i)] \in \mathbf{R}^{\mathbf{D}}$ is the external input data to the low level minimization process. It can be either the original range image or an estimated first derivative image from the first phase.

The closeness term measures the closeness of the solution f and the input d

$$E^c(f \mid d) = \sum_{i \in \mathbf{D}} [f(i) - d(i)]^2$$

The smoothness term measures the smoothness of the solution f with respect to the neighborhood system ω which will be defined shortly. This term is defined by

$$E^s(f \mid \pi, \omega) = \sum_{i \in \mathbf{D}} \sum_{j \in \omega(i)} \pi(i,j)[f(i) - f(j)]^2$$

At this level, the smoothness term can be regarded as a measurement of the second order discrepancy. The neighborhood system ω is defined by $\omega = \{\omega(i) \mid j \in \mathbf{D}\}$ where

$$\omega(i) = \{j \mid 0 < (i-j)^2 \le r\} \tag{6}$$

is the set of neighboring pixels of i in which r is an integer concerning the radius of the neighborhood. Note that $i \notin \omega(i)$. We choose $r = 1$ at this level such that $\omega(i) = \{i-1, i+1\}$

consists of i's adjacent neighbors. The function π determined the connection between nodes in a computational network. It is defined by

$$\pi(i,j) = g(\rho(i,j)/\sigma_2) \tag{7}$$

In the above, $g(x)$ is an even function of x given parameter σ_2. It takes the maximum when $x = 0$, monotonically decreases as $|x|$ increases and approaches to 0 as $x \to \infty$. Possible choices of the function g include

$$g(x) = e^{-|x|}$$

and

$$g(x) = 1/(1 + x^2)$$

And $\rho(i,j)$ is a defined distance between pixel values $f(i)$ and $f(j)$

$$\rho(i,j) = \begin{cases} 0 & \text{if } \exists j \in \omega(i) \text{ such that } |d(i) - d(j)| \leq \alpha \\ +\infty & \text{otherwise} \end{cases} \tag{8}$$

where α is an threshold for detecting discontinuities (a discussion about selecting α will be given shortly). The connection $\pi(i,j) = 1$ or 0 and $\pi(i,j) = \pi(j,i)$ is symmetric. We choose a finite σ_2 for the function g. Therefore, wherever there is a discontinuity between i and j, the defined distance $\rho(i,j) = +\infty$ such that $\pi(i,j) = 0$ dis-enables interaction between the two points; otherwise $\rho(i,j) = 0$ such that $\pi(i,j) = 1$ enables the interaction. Hence the continuity-control is embedded into the connection π.

We have a general definition $\pi(i,I;j,J) = g(|\rho(i,j) - \rho(I,J)|/\sigma_2)$ as will be defined in the following subsections for the intermediate and high levels. In this definition, I and J are "points" in a model image and $\rho(I,J)$ is the distance between them. The model can be implicit or explicit. At the high level, the model is explicitly determined by a model object. At this level as well as the intermediate level, the model is implied by the smoothness assumption. This is the model of flatness; every pixel in this model image takes the same value with one another, and thus the distance between two "points" in this model is always zero, expressed by $\rho(I,J) \equiv 0$. Image points i and j are to be mapped to model points I and J, respectively. This results in mapping from $d(i)$ and $d(j)$ to $f^*(i)$ and $f^*(j)$. This will be further discussed in the next subsection. At this level, we have simply $\pi(i,j) = g(|\rho(i,j) - 0|/\sigma_2)$.

There are always questions about thresholds. Ideally, the threshold α in Eq. (8) must be somehow related to the scale of data and to the level of noise. Nevertheless, accurate detection of discontinuities is not crucial here for the invariant surface segmentation. The inadequacy of the low level processing (the derivative and curvature estimation) can be compensated by the intermediate level (see the next subsection) which significantly improves the coherence of the segmentation. False or missing detection of some of the discontinuities will not cause the failure of the whole process. Therefore we have used a simple scheme in determining α [62].

However, better detection of discontinuities is always desired. Some algorithms by other researchers may lead to improvements of the quality of the derivative estimation, such as Blake-Zisserman's GNC algorithm [13] and an algorithm of generalized smoothing networks by Harris

[38] and Liu and Harris [66]. The former does surface reconstruction and detects discontinuities in a single process, and has yielded good performance. The latter does surface reconstruction, detects discontinuities and estimates the first and the second derivatives in a single network, which is very attractive for the curvature-based segmentation we aim to perform.

3.2 Intermediate level

The computation is done on a domain $\mathbf{D} = \{(i, I) \mid 1 \leq i \leq m, 1 \leq I \leq 3\}$ where m is the size of the image and $I = 1, 2, 3$ represents, respectively, the negative, zero, positive sign of curvature. Here $\mathbf{D}$ is the cross product of the image lattice $\{1, ..., m\}$ and the set of the sign labels $\{1, 2, 3\}$. The problem here is to classify each pixel into one of the labels $\{1, 2, 3\}$ rather than a reconstructed surface value as in the low level. We use a function $f = [f(i, I)] \in \{0, 1\}^{\mathbf{D}}$ to represent the solution, with $f(i, I) = 1$ or 0 representing whether or not i is labeled I. The function f is subject to the *consistency constraint*

$$\sum_I f(i, I) = 1 \quad \forall i \tag{9}$$

This constrains each pixel point to being assigned one and only one label. The final f^* is a refined image labeling map. The function $d = [d(i, I)] \in \{0, 1\}^{\mathbf{D}}$ is the external input data to the intermediate level minimization process. It is obtained from a coarse curvature sign map $s = [s(i)]$

$$d(i, I) = \begin{cases} 1 & \text{if } I = s(i) \\ 0 & \text{otherwise} \end{cases}$$

where $s(i)$ is the curvature sign at pixel i. The coarse map s is obtained by thresholding either an estimated H image or K image at appropriate thresholds $\pm\tau$

$$s(i) = \begin{cases} 3 & \kappa(i) > \tau \\ 2 & |\kappa(i)| \leq \tau \\ 1 & \kappa(i) < -\tau \end{cases} \tag{10}$$

where κ is a curvature image and 3, 2 and 1 represent, respectively, the positive, zero and negative curvature sign. A discussion of the threshold τ will be given later in this section. Examples of s have been shown in the first two maps at the middle rows of Figures 1 and 2.

The closeness term again measures the closeness of the solution f and the input d

$$E^c(f \mid d) = \sum_{(i,I)\in\mathbf{D}} [f(i, I) - d(i, I)]^2$$

By expanding the left-hand side of the above, noting that f satisfies the consistency constraint and dropping resulting constants without affecting the minimization, we can simplify E^c into the following

$$E^c(f \mid d) = -2 \sum_{(i,I)\in\mathbf{D}} d(i, I) f(i, I)$$

This is is similar to the criterion for the simplest *perceptron* [82] and also to the cost energy in a *minimal mapping theory* [91]. The smoothness term is defined by

$$E^s(f \mid \pi, \omega) = -2 \sum_{(i,I)\in\mathbf{D}} \sum_{(j,J)\in\omega(i,I)} \pi(i,I;j,J) f(i,I) f(j,J)$$

In the above, the neighborhood system ω is defined by $\omega = \{\omega(i,I) \mid (i,I) \in \mathbf{D}\}$ where

$$\omega(i,I) = \{(j,J) \mid 0 < (i-j)^2 \leq r, J = I\} \tag{11}$$

is the set of neighboring pixels of i within the radius of $\sqrt{r}$ which are currently labeled I. Note that $(i,I) \notin \omega(i,I)$. At this level, r is related to a noise measurement V on the input image (see Section 6); the larger V, the larger r. The function π is the connection between elements of a computational network. It is defined by

$$\pi(i,I;j,J) = g(|\rho(i,j) - \rho(I,J)|/\sigma_2) \tag{12}$$

where the function $g(\cdot)$ is as that defined at the low level and $\rho(i,j)$ is determined by the low level and therefore is identical to that which determines depth discontinuities in the range image. In defining π above, the distance function $\rho(I,J)$ is related to an inherent relationship among three curvature sign labels

$$\rho(I,J) = \begin{cases} 0 & \text{if } I = J \\ +\infty & \text{otherwise} \end{cases}$$

The interpretation of the above is such that the distance is 0 within the same label and is ∞ between different labels. The connection $\pi(i,I;j,J) = 1$ or 0 depends on $|\rho(i,j) - \rho(i,J)|$. In fact the above definition of $\pi(i,I;j,J)$ is equivalent to the following

$$\pi(i,I;j,J) = \begin{cases} \pi(i,j) & \text{if } I = J \\ 0 & \text{otherwise} \end{cases}$$

where $\pi(i,j)$ is the connection between i and j of the low level and is determined by the depth discontinuity operation in Eq. (8). Hence the continuity-control is embedded into the connection π. Note the symmetry $\pi(i,I;j,J) = \pi(j,J;i,I) = \pi(j,I;i,J) = \pi(j,I;i,J)$. The smoothness term at this level has a different meaning from that at the low level. The original idea [61] of this term is meant to perform a multi-level morphological closing operation [85] by optimizing a coherence measure. Its negative can be interpreted as the *average local consistency* with respect to the neighborhood system ω, in relaxation labeling [49].

Now let us go back to the end of the previous subsection and compare the interpretation we have made of $\rho(I,J)$ here, to that of the flatness model at the low level. At this level, pixel i is mapped to a label I which takes a discrete value in $\{1,2,3\}$. At the low level, every i can be though of as being mapped to the implicit model value "$I'' = f(i)$ which takes a real. The flatness model is such that the value of every "pixel" is equal to every other pixel. Therefore, we have at the low level $\rho(I,J) \equiv 0$, and $\pi(i,I;j,J) = g(|\rho(i,j) - 0|/\sigma_2)$ or $\pi(i,j) = g(|\rho(i,j)|/\sigma_2)$.

The threshold τ for thresholding the estimated curvature images is determined by histogram analysis. The curvature images we obtained are quite smooth, as can be seen from the histograms of the curvature images at the upper-right corners of Figures 1, 2. There are clear peaks and valleys in the image histograms. In this situation, threshold selection is not a burden. For the mean curvature images, it is fixed at $\tau = 0.006$. For the Gaussian curvature images, it is one of the two candidate values, $\tau_1 = 0.0008$ and $\tau_2 = 0.00006$ at which the first and the second valleys appear in the image histogram. The decision between the two is made according to the a rule based on the morphological analysis of the curvature image histograms [62]. This scheme has worked for more than a dozen of images containing objects of different complexity.

3.3 High level

We assume that the 3D surface matching problem has been reduced to that of inexact sub-ARG matching. This problem is more difficult than the general subgraph matching. In the first place, there can be nodes in the scene ARG which has no counterparts in the model ARG, *e.g.* due to noise or mal-segmentation. Secondly, a node in the model ARG can be split into many. To deal with the first problem, we will wish to classify nodes due to mal-segmentation as $NULL$. The second problem is handled by allowing many-to-one matching.

We are given a scene ARG $\mathbf{g} = (v, a)$ and a model ARG $\mathbf{G} = (V, A)$. Their nodes are indexed by $\{1, ..., m\}$ and $\{0, 1, ..., M\}$, respectively. The number 0 in the model ARG indices the $NULL$ node to which patches in the scene ARG emerging due to noise should be matched. The computation can be performed, in principle, on a labeling domain formed from the cross product $\{1, ..., m\} \times \{0, 1, ..., M\}$. However, plausible matches are confined to a subset of this domain by some symbolic unary constraints. The first is $hk(i) = HK(I)$ which means that both the scene patch and the model patch must have the same surface type in order to be matched. The second is $area(i) \leq 4 \times AREA(I)$. Strictly speaking, this second condition should be $area(i) \leq AREA(I)$ under any occlusion, but we include a factor of 4 to allow a looser constraint to tolerate segmentation errors due to noise. Using these symbolic constraints, we can screen out symbolically implausible matches to reduce the complexity of the search space for the combinatorial optimization problem in order that correct solutions are easier to find using relaxation method. Now we can define the domain of surface matching to be $\mathbf{D} = \{(i, I) \mid v^1(i) = V^1(I), v^2(i) \leq 4 \times V^2(I), 1 \leq i \leq m, 0 \leq I \leq M\}$ where $a^1 = hk$ and $a^2 = area$ are the attributes of scene nodes, and and $A^1 = HK$ and $A^2 = AREA$ are those of model nodes (refer to section 2.3). By assumption, $A^k(0)$, the attributes of model node 0, are "wildcards" such that and $hk(i) = HK(0)$ and $area(i) \leq 4 \times AREA(0)$ are always true. Note that at this level, $\mathbf{D}$ may not be a homogeneous lattice as are at the previous two levels.

We use function $f = [f(i, I)] \in [0, 1]^{\mathbf{D}}$ to denote the state of surface patch labeling, with $f(i, I) \in [0, 1]$ representing the plausibility that i is mapped to I, subject to the consistency constraint

$$\sum_I f(i, I) = 1 \qquad \forall i$$

The final f^* is the surface matching result and is required to be *unambiguous*

$$f^* \in \{0, 1\}^{\mathbf{D}} \qquad \forall i, I$$

with $f^*(i,I) = 1$ meaning that scene patch i is unambiguously matched to model patch I and $f^*(i,I) = 0$ not matched. The external input data $d = [d(i,I)] \in \{0,1\}^{\mathbf{D}}$ is computed based on attributes of image patch i and model patch I

$$d(i,I) = \begin{cases} 1 & \text{if } (i,I) \in \mathbf{D} \\ 0 & \text{otherwise} \end{cases}$$

The above definition is symbolic. It takes account only of symbolic constraints $hk(i) = HK(I)$ and $area(i) \leq 4 \times AREA(I)$. It can also be defined to take a real value in $[0,1]$. For example, if the attributes of the average of curvature (attributes No.3 and 4 in Section 2.3) are used the input can be defined by

$$d(i,I) = \begin{cases} \exp\{-\sum_{k=3}^{4} |v^k(i) - V^k(I)|/\sigma_1^k\} & \text{if } (i,I) \in \mathbf{D} \\ 0 & \text{otherwise} \end{cases}$$

where σ_1^k is are control parameters for encoding unary constraints $k = 3,4$.

The closeness term measures the closeness of the solution f and the input d. At this level, it take the same form as that at the intermediate level

$$E^c(f \mid d) = \sum_{(i,I) \in \mathbf{D}} [f(i,I) - d(i,I)]^2$$

By expanding the left-hand side of the above, noting that f satisfies the consistency constraint and dropping resulting constants without affecting the minimization, we can simplify E^c into the following

$$E^c(f \mid d) = -2 \sum_{(i,I) \in \mathbf{D}} d(i,I) f(i,I)$$

The smoothness term at this level takes the same form as that at the intermediate level

$$E^s(f \mid \pi, \omega) = \sum_{(i,I) \in \mathbf{D}} \sum_{(j,J) \in \omega(i,I)} \pi(i,I;j,J) f(i,I) f(j,J)$$

In the above, the neighborhood system ω is defined by $\omega = \{\omega(i,I) \mid (i,I) \in \mathbf{D}\}$ where

$$\omega(i,I) = \{(j,J) \mid 0 < |\rho(i,j) - \rho(I,J)| \leq r, (j,J) \in \mathbf{D}\} \quad (13)$$

Here $\rho(i,j) = a^1(i,j)$ is the minimum distance between patches i and j of the scene ARG and $\rho(I,J) = A^1(I,J)$ is that between I and J of the model ARG. Therefore $\omega(i,I)$ is the set of neighboring points of (i,I) in $\mathbf{D}$ which satisfy that the relational discrepancy $|\rho(i,j) - \rho(I,J)|$ is smaller than r but nonzero. Note that $(i,I) \notin \omega(i,I)$. At this level, we choose $r = +\infty$ such that $\omega(i,I)$ virtually includes all points in $\mathbf{D}$ except for (i,I). The smoothness term has a different meaning from that at the low level. Its negative actually measures the the *average local consistency* [49].

The function π is the connection between nodes of a computational network. It is defined by

$$\pi(i,I;j,J) = g\left(\sum_{k=1}^{K2} |a^k(i,j) - A^k(I,J)|/\sigma_2^k\right) \qquad i \neq j \quad (14)$$

where $K2$ is the number of binary relations, σ_2^k is a parameter for encoding the binary constraint. and the function $g(\cdot)$ is the same as that defined at the low level (*e.g.* $g(x) = e^x$). Here $K2$ is 1 since we use only one source of binary constraint, the minimum distance between patches. For $NULL$ matches, we define

$$\pi(i, I; j, 0) = \pi(i, 0; j, J) = \pi_{2,0} \tag{15}$$

where $0 \leq \pi_{2,0} \leq 1$ is a constant. The higher $\pi_{2,0}$, the more likely scene nodes are mapped to $NULL$. If $\pi_{2,0} = 0$, then none of the scene nodes will be so. Note $0 < \pi(i, I; j, J) \leq 1$ and the symmetry $\pi(i, I; j, J) = \pi(j, J; i, I) = \pi(j, I; i, J) = \pi(j, I; i, J)$.

The definition of the connection π in Eq. (14) can be modified to include symbolic relations. Suppose we have $K2$ types of relations denoted by $a^k(i, j)$ for the scene and $A^k(I, J)$ for the model where $k = 1, ..., K2$. We can encode symbolic constraints into the connection by appropriately choosing σ_2^k (and similarly σ_1^k for unary constraints). Suppose, for instance, the p^{th} relation is symbolic, with $a^p(i, j) = 1$ representing its presence between nodes i and j and 0 its absence. If $a^p(i, j) \neq A^p(I, J)$, then we choose a very small $\sigma_2^p = 0^+$ such that $|a^p(i, j) - A^p(I, J)|/\sigma_2^p = +\infty$ and hence $\pi(i, I; j, J) = 0$, regardless of whether other binary constraints are satisfied; otherwise if $a^p(i, j) = A^p(I, J)$, then $|a^p(i, j) - A^p(I, J)|/\sigma_2^p = 0$ and $\pi(i, I; j, J)$ is determined by the remaining types of relations. In this way, both numerical and symbolic constraints can be embodied into connections.

The binary relations $a^k(i, j)$ and $A^k(I, J)$ discussed above are symmetrical, *i.e.* $a^k(i, j) = a^k(j, i)$, *etc.* There can be anti-symmetrical relations, *e.g.* above/below, inside/outside, *etc.* Supposing binary relation k is about right-left-ness, we can use $a^k(i, j) = 1$ to represent the relation that the i is left to j, $a^k(i, j) = 0$ to represent the reverse, and choose a small σ_2^k (> 0) to embed this symbolic constraint into the connection. In this situation, the symmetry property of the connection no longer holds, but this has no unfavorable effect on the performance of minimization process.

In our formulation, many-to-one matches are allowed More than one nodes in the scene ARG can be matched to a node in the model ARG at the same time. However there cannot be one-to-many unambiguous matches. This is because of the consistency constraint $\sum_I f(i, I) = 1$.

3.4 Discussion

The computational engine at the low level maps a pixel i to a reconstructed surface value $f^*(i) \in \mathbf{R}$. The mapping is based on the input d (which can be either the original range image or an estimated first derivative image) and on the connection π computed from the range image. The final result of this level is an estimated H and K image computed from the five derivative estimates. The computational engine at the intermediate level maps a pixel i to a curvature sign $I \in \{1, 2, 3\}$. The mapping is based on the input d, which in this case can be either a coarse H or a coarse K map produced by a thresholding operation, and on the connection π computed from the range image. The final result of this level are the refined H and K maps from which the HK can be computed. The continuity-control mechanism at these two levels is embedded into the connection π. The computational engine at the high level maps a scene patch $i \in \{1, ..., m\}$ to a model patch $I \in \{0, ..., M\}$. The mapping is based on the input d and connection π. Both d and π are computed from the scene ARG and a model ARG.

The optimization at the low level is unconstrained whereas that at the other two levels is constrained. The former maps a point i to $f^*(i)$, a point in the continuous set of real, whereas the latter maps i to I, a point in a discrete set of labels. The result of the former is, in principle, piecewise continuous whereas that of the latter is combinatorial. For the convenience of computation, we use a labeling map $\{f(i, I)\}$ to denote the state of combinatorial labeling for the latter. In fact we could also use $f(i) \in \mathbf{D}_I$ to denote the label i is currently assigned, where $\mathbf{D}_I = \{1, 2, 3\}$ for the intermediate level or $\mathbf{D}_I = \{0, 1, ..., M\}$ for the high level, for a discrete updating scheme. We use the labeling map because, for computational reasons, the continuous approach to a combinatorial problem is sometimes needed to find better solutions as is the case for the high level. A small difference exists between the formulation of the intermediate level and that of the high level. At the high level, there is a label of unknown, that is, $NULL$ or 0 in $\mathbf{D}_I$. There is some freedom in defining connection, or compatibility, in relation to the $NULL$ label.

A merit of our high level work is the numerical definition of the connection (compatibility) function π of the matching network. Compatibilities have been defined in previous work for matching two relational structures, *e.g.* in [2, 22, 52, 73] [81, 86, 96]. However, all their definitions involve *thresholds*. These thresholds are used to determine whether or not two binary relations are compatible depending on whether or not the difference of two numerical relations is smaller than a threshold; or they are used to determine whether or not there is a connection between two nodes of a matching (labeling) network depending on whether or not the computed compatibility is larger than the threshold. This is a symbolic decision. The symbolic determination can cause potential failure of the matching if some of the symbolic compatibilities are satisfied with respect to the thresholds. In contrast, the parameters σ_2 in our definition of π are not meant to threshold; rather they provide a flexible control of the connection. The control can be "hard" (symbolic) or "soft" (numerical). Our definition embodies both symbolic constraints and numerical ones into the network connection in a unified way. Furthermore, the formulation can encode constraints of orders higher than two as easy as constraints of lower orders. This is a bonus since high order constraints are necessary for some problems [56].

The whole matching and recognition process at the high level consists of N matching phases if there are N models. The best matched model, that with the least remaining energy, is chosen as the recognition result. However, more than one model ARGs can be built into a *universal model* ARG [64]. The matching then consists of only one phase: from the scene to the universal model; overlapping parts of different objects are segregated at the same time of matching.

As has been seen, the high level processing is almost identical to that at the intermediate level. The high level task are model-based and cast as ARG matching. This, together with the discussions at the end of the previous two subsections about the relations between the first two levels, suggests that there are more inherent similarities and parallels among the three levels. We can give an interpretation of regularization techniques from a new perspective, that in terms of graph matching. Regularizing can be interpreted as inexact matching from images to an implicit relational model graph. The model there is that of flatness where the binary relation is $\rho(I, J) = 0$ everywhere.

4 Minimization Algorithms

The minimization at the low level is unconstrained whereas that at the other two levels has to satisfy the consistency constraint. We use relaxation algorithms for the computation of minimal solutions. A solution is initialized based on data d and the updated iteratively by using some rule until it converges.

4.1 Low Level

We are finding a point f^* in space $\mathbf{S} = \mathbf{R}^{\mathbf{D}}$, where $\mathbf{D} = \{i \mid 1 \le i \le m\}$, which minimizes

$$E(f \mid d, \pi, \omega) = \sum_{i \in \mathbf{D}} [f(i) - d(i)]^2 + \lambda \sum_{i \in \mathbf{D}} \sum_{j \in \omega(i)} \pi(i,j)[f(i) - f(j)]^2$$

To compute the minimum, we introduce a time parameter into f such that

$$f = f^{(t)}$$

We are interested in constructing a dynamic system that can locate as the final solution a minimum of the energy function. We require that the energy E should not increase along its trajectory $f^{(t)}$ as the system evolves, *i.e.* $\frac{dE}{dt} \le 0$. A simple dynamic system with characteristic $\frac{dE}{dt} \le 0$ and $\lim_{t \to \infty} \frac{dE}{dt} = 0$ is determined by a system of differential equations

$$-\mu \frac{df(i)}{dt} = q(i) \qquad \forall i \tag{16}$$

where $\mu > 0$ is a constant small enough to ensure the convergence of the algorithm and

$$q(i) = \frac{1}{2} \frac{\partial E}{\partial f(i)} = [f(i) - d(i)] + 2\lambda \sum_{j \in \omega(i)} \pi(i,j)[f(i) - f(j)]$$

is a component of $\frac{1}{2} \nabla E(f)$. The energy change due to changes in f is

$$\frac{dE}{dt} = \sum_i \frac{\partial E}{\partial f(i)} \frac{df(i)}{dt} = -\frac{2}{\mu} \sum_i [q(i)]^2 \le 0$$

The system will converge to *the* minimum energy state, $f^* = f^{(\infty)}$, whatever the initial state is set to because E is quadratic with π fixed. Once initialized, the system will evolve to settle down to the minimum energy state, that of the least power dissipation. Note that there are two trivial solutions. The first is $f^* = d$ when $\lambda = 0$ such that only the closeness term contributes to the energy function. This is the *minimum squared error* solution. The second is some piecewise constant function due to unlimited diffusion smoothing when $\lambda \to +\infty$ such that only the smoothness term contributes. We are interested in non-trivial solutions where $0 < \lambda < +\infty$.

Eq. (16) describes the dynamics of an analog neural network which will be discussed later. For discrete implementation, the computation is done according to the following updating rule

$$f^{(t+1)}(i) = f^{(t)}(i) \ - \ \frac{1}{\mu} q^{(t)}(i)$$

In the 2D image case, the computation is done on the image grid $\mathbf{D} = \{(i,j) \mid 1 \leq i,j \leq m\}$. The 2D extension to Eq. (16) is

$$-\mu \frac{df(i,j)}{dt} = q(i,j) \qquad\qquad \forall i,j \tag{17}$$

where

$$q(i,j) = \frac{1}{2} \frac{\partial E}{\partial f(i,j)} = [f(i,j) - d(i,j)] + 2\lambda \sum_{(u,v)\in\omega(i,j)} \pi(i,j;u,v)[f(i,j) - f(u,v)]$$

$$\omega(i,j) = \{(u,v) \mid 0 < (u-i)^2 + (v-j)^2 \leq r\}$$

$$\pi(i,j;u,v) = g(|\rho(i,j;u,v) - 0|/\sigma_2)$$

$$\rho(i,j;u,v) = \begin{cases} 0 & \text{if } \exists(u,v) \in \omega(i,j) \text{ such that } |d(i,j) - d(u,v)| \leq \alpha \\ +\infty & \text{otherwise} \end{cases}$$

It is not difficult to transform Eq. (17) into a discrete updating rule

$$f^{(t+1)}(i,j) = f^{(t)}(i,j) \ - \ \frac{1}{\mu} q^{(t)}(i,j) \tag{18}$$

4.2 Intermediate Level

We are finding a point f^* in space $\mathbf{S} = \{f(i,I) \mid f(i,I) \in \{0,1\},\ \sum_I f(i,I) = 1,\ (i,I) \in \mathbf{D}\}$, where $\mathbf{D} = \{(i,I) \mid 1 \leq i \leq m, 1 \leq I \leq 3\}$, which minimizes

$$E(f \mid d,\pi,\omega) = -2 \sum_{(i,I)\in\mathbf{D}} d(i,I)f(i,I) - 2\lambda \sum_{(i,I)\in\mathbf{D}} \sum_{(j,J)\in\omega(i,I)} \pi(i,I;j,J)f(i,I)f(j,J)$$

To compute the minimum, we again introduce a time parameter into f such that

$$f = f^{(t)}$$

We are interested in constructing a dynamic system that can locate as the final solution a minimum of the energy function. We require that the energy E should not increase along its trajectory $f^{(t)}$ as the system evolves, *i.e.* $\frac{dE}{dt} \leq 0$.

There are two approaches to solutions for the unambiguous labeling problem. The first is to find combinatorially a point in the above space which minimizes the energy. This is referred to as discrete relaxation. The second is continuous relaxation. In this approach, one starts with a point in space $\mathbf{S}^+ = \{f(i,I) \mid f(i,I) \in [0,1],\ \sum_I f(i,I) = 1,\ (i,I) \in \mathbf{D}\}$ (cf. $\mathbf{S}$) of which $\mathbf{S}$ consists of corners, and approach an unambiguous solution, namely a minimum in $\mathbf{S}$, as the system evolves. The continuous approach usually requires more real computation and is therefore more expensive, but can in general lead to better results because the updating therein is performed in a more careful and gradual way. In this work, we use the discrete approach for the computation at the intermediate level but the continuous one for the high level. At this level, the initial solution $f^{(0)}$ is assigned according to a coarse curvature sign map, which is quite close to a global solution. Satisfactory results can be obtained by using a simple discrete updating rule. However at the high level, it is not so easy to get a good initial estimate for the solution and thus a more sophisticated algorithm has to be used to find better solutions.

Denote the local energy change at pixel i at time step t due to local state change $\{\Delta f^{(t)}(i,I) \mid 1 \leq I \leq 3\}$ by $\Delta E^{(t)}(i)$. The change $\Delta f^{(t)}(i,I)$ takes a value of -1, $+1$ or 0 since $f^{(t)}(i,I)$ can change from 1 to 0, vice versa or remain unchanged. The local energy change is

$$\Delta E^{(t)}(i) = 2\sum_I q^{(t)}(i,I)\Delta f^{(t)}(i,I)$$

where

$$q^{(t)}(i,I) = -d(i,I) - 2\lambda \sum_{(j,J)\in\omega(i,I)} \pi(i,I;j,J) f^{(t)}(j,J)$$

is a component of $\frac{1}{2}\nabla E$. Meanwhile, we have a constraint for the state changes $\sum_I \Delta f^{(t)}(i,I) \equiv 0$ because of the consistency constraint $\sum_I f(i,I) \equiv 1$. We update $f^{(t)}(i,I)$ according to the following rule

$$\begin{array}{c} f^{(t+1)}(i,S) \leftarrow 1 \\ f^{(t+1)}(i,R) \leftarrow 0 \\ \text{if} \\ q^{(t)}(i,S) < q^{(t)}(i,R) \end{array}$$

$\forall R \neq S$

where the updating is performed *in parallel* for all pixels $i \in \{1,...,m\}$. Node (i,S) is activated $f(i,S) \leftarrow 1$ if the gradient is the least at (i,S), or in other words if node (i,S) receives maximum support, and the rest are suppressed $f(i,R) \leftarrow 0$, $\forall R \neq S$. Suppose changes happen at (i,S) and (i,R'), that is, $f^{(t)}(i,S)$ is changed from 0 to 1 and $f^{(t)}(i,R')$ is changed from 1 to 0. Therefore $\Delta f^{(t)}(i,S) = 1$ and $\Delta f^{(t)}(i,R) = -1$ and

$$\Delta E^{(t)}(i) = 2[q^{(t)}(i,S) - q^{(t)}(i,R')]$$

is always negative since $q^{(t)}(i,S) < q^{(t)}(i,R')$. This means that the updating rule ensures that the minimization process always converges to a minimum state after a sufficient number of steps, *i.e.* $f^* = f^{(\infty)}$, because the energy E is lower-bounded and the updating is always performed in such a way as to reduce local energies.

When the weight $\lambda = 0$, only the closeness term contributes to the energy E and the minimization solution is identical to the thresholding result in Eq. (10), *i.e.* $f^* = d$. When λ is sufficiently large, the smoothness term in $E(f^{(t)})$ dominates. In this situation, computing gradient is based merely on counting numbers regardless of the external input d

$$q^{(t)}(i,I) = -2\lambda\#\{(j,J) \mid f(j,J) = 1, \pi(i,I;j,J) = 1, (j,J) \in \omega(i,I)\}$$

where # is the cardinality of a set, The updating rule then becomes the local minimum selection.

While the energy function at the low level is convex, the energy functions at this and the high levels are generally non-convex. There are many local minima in the solution space. The solution depends on the algorithm used for the minimization and the initial configuration $f^{(0)}$. We set the initial state to the external input $f^{(0)} = d$ and $\lambda = +\infty$. Then we update $f^{(t)}$ according to the rule until the iteration converges, *i.e.* $f^{(t+1)} = f^{(t)}$. In other words, we virtually set $\lambda = 0$ for the first iteration and switch λ to $+\infty$ for the rest of iterations. Here a local minimum close to $f^{(0)}$, which is not necessarily the global one, is usually good enough for our purpose.

Figure 7: ===== about here =====

Oscillations may occur in the discrete updating. For this reason, we use an updating scheme that is a hybrid of deterministic and stochastic approaches, which we shall call a *randomized gradient descent method*. The randomness is introduced to achieve better results and to prevent possible oscillation during the minimization process while the computation is still very cheap compared with stochastic methods such as simulated annealing [1, 31, 54]. The randomness is implemented by randomly choosing one neighborhood from two candidate ones as $\omega(i)$ during the computation of $q(i, I)$. Figure 7 illustrates the two candidates of the neighborhood point set where the neighborhood size parameter r is 5. Which one to use is determined by an evenly-distributed random number in the set of {0,1}.

In the 2D image case, the computation is done on the image grid $\mathbf{D} = \{(i,j) \mid 1 \le i,j \le m\}$. The 2D extension to the 1D formulation can be obtained by substituting i, j for i and u, v for j in the above discussion, and modifying the definitions of ω, ρ and π accordingly.

Examples of refined H, K and HK maps have been shown at the bottom of Figures 1 and 2. The results demonstrate great improvement in comparison with those produced by the previous surface fitting techniques, *e.g.* [7, 27, 37, 43, 61, 97], for free-formed surfaces in particular. For surface fitting methods, the most difficult problem is determining the order of the models. Low order models are generally used, which greatly limits the applicability of the resulting algorithms. More specifically, algorithms based on the fitting paradigm work quite well if the models are correct, otherwise they fail. Using variable order can improve the situation but does not solve the problem adequately. This can be seen from the segmentation results for the Renault part and the torus images [5] whose order is high and less predictable. In contrast, our work makes no explicit assumption about the order of surfaces. The only assumption made is the "smoothness" which puts far less restriction and allows wider applicability. Another problem is, as widely reported by other researchers, that the smoothed images are often blurred near discontinuities. In our work, the introduction of the depth-discontinuity-controlled regularizers effectively eliminates the blurring effect at *depth* discontinuities. Algorithmically, our work involves less parameters and the chosen values of parameters are applicable to a rather large class of surface [62]. However, there is a problem in our formulation: minimizing the energy tends to round crease edges, as shown in Figure 8. This is because only *depth* discontinuities are encoded into the energy function while *orientation* discontinuities are ignored. The problem would be avoided if a rough estimate of orientation discontinuities could be done before the minimization process and added into the controller π.

Figure 8: ===== about here =====

4.3 High Level

We are finding a point f^* in space $\mathbf{S} = \{f(i,I) \mid f(i,I) \in \{0,1\},\ \sum_I f(i,I) = 1,\ (i,I) \in \mathbf{D}\}$, where $\mathbf{D} = \{(i,I) \mid hk(i) = HK(I), area(i) \leq 4 \times AREA(I), 1 \leq i \leq m, 0 \leq I \leq M\}$, which minimizes

$$E(f \mid d,\pi,\omega) = -2 \sum_{(i,I)\in\mathbf{D}} d(i,I)f(i,I) - 2\lambda \sum_{(i,I)\in\mathbf{D}} \sum_{(j,J)\in\omega(i,I)} \pi(i,I;j,J)f(i,I)f(j,J)$$

To compute the minimum, we again introduce a time parameter into f such that

$$f = f^{(t)}$$

We are interested in constructing a dynamic system that can locate as the final solution a minimum of the energy function. We require that the energy E should not increase along its trajectory $f^{(t)}$ as the system evolves, *i.e.* $\frac{dE}{dt} \leq 0$.

The simple discrete updating algorithm used for the previous level finds correct solutions for the problem at this level when the scene ARG is close to a sub-part of the model ARG. Unfortunately, it fails to find satisfactory solutions when there are serious distortions between the scene ARG and any sub-part of the model ARG. Its performance could be improved in principle if the updating rule is modified into a stochastic version [1, 31, 54]. However, we use instead the continuous relaxation labeling [26, 49]. Using this method, an optimal constrained combinatorial solution in the discrete space $\mathbf{S}$ is approached via the continuous space $\mathbf{S}^+ = \{f(i,I) \mid f(i,I) \in [0,1], \sum_I f(i,I) = 1,\ (i,I) \in \mathbf{D}\}$ where the space $\mathbf{S}^+$ is a subspace of the hypercube of $\mathbf{S}$. The following is an outline of the relaxation labeling algorithm based on constrained gradient descent.

Algorithm: Relaxation Labeling
Given: d, π and $f^{(0)}$
Output: Unambiguous $f^* = f^{(\infty)}$
Begin
  Set $t \leftarrow 0$
  Do
    Compute $q^{(t)}$ from d, π and $f^{(t)}$
    Update $f^{(t+1)} \leftarrow \Psi(f^{(t)}, q^{(t)})$
    $t \leftarrow t+1$
  Until (converged)
  Set $f^* \leftarrow f^{(t)}$
End of Algorithm

In the above, $q^{(t)}$ is the gradient

$$q^{(t)}(i,I) = -d(i,I) - 2\lambda \sum_{(j,J)\in\omega(i,I)} \pi(i,I;j,J) f^{(t)}(j,J)$$

and Ψ is some updating operation. The process starts with an initial state $f^{(0)}$. At time t, gradient function $q^{(t)}$ is computed and the state $f^{(t)}$ is updated to reduce the energy using an updating rule

$$f^{(t+1)} = \Psi(f^{(t)}, q^{(t)})$$

At the intermediate level, this updating operation Ψ selects and activates the node with the least gradient and suppresses the rest. Here in continuous relaxation labeling, it is meant to update f by an appropriate amount in the appropriate direction. We use the gradient projection operation described in [49, 72] for Ψ. (It should be noted that their original goal was to maximize a criterion by gradient ascent. Therefore, the direction of gradient projection should be reversed for our minimization purpose). This operation computes the updating direction u. Then f is updated by a vector of ηu

$$f^{(t+1)} \leftarrow f^{(t)} + \eta u$$

where u is the direction vector computed by the projection operation and η is a scale factor which ensures that the updated vector $f^{(t+1)}$ lies within the space S^+. In practice, we use the modified version of their algorithm [49]

$$f^{(t+1)}(i) \leftarrow f^{(t)}(i) + \eta_i u(i)$$

which updates each $f(i)$ individually to simplify the computation. The iteration terminates when the algorithm converges, *i.e.* $f^{(t+1)} = f^{(t)}$. The final labeling is in general unambiguous. The unambiguity means that each node has a single interpretation. For details of the algorithm and its convergence properties, readers are referred to [49].

We set the initial state $f^{(0)}(i,I)$, $(i,I) \in \mathbf{D}$, according to the following

$$f^{(0)}(i,I) = \begin{cases} \frac{1+\delta(i,I)}{\sum_{I:(i,I)\in\mathbf{D}}[1+\delta(i,I)]} & \text{if } (i,I) \in \mathbf{D} \\ 0 & \text{otherwise} \end{cases}$$

where $\delta(i,I)$ is a small random deviation which makes $f^{(0)}(i,I) \neq f^{(0)}(i,J)$, $\forall I \neq J$, $(i,I),(i,J) \in \mathbf{D}$. We set the weight $\lambda = +\infty$; ¿In other words, we virtually set $\lambda = 0$ for the first iteration and switch λ to $+\infty$ for the rest of iterations. The updating is parallel for all the scene nodes $i \in \{1, ..., m\}$.

Figure 9: ===== about here =====

Table 2: ===== about here =====

In the following, we present two experiments to demonstrate the inexact ARG matching. We generate a simulated model ARG as shown in Figure 9a. The ARG has 40 nodes. Each of these

Table 3: ===== about here =====

nodes is attributed by one of three different types, shown in different sizes. The type attribute is a simulation of HK labels. Between each pair of nodes is a numerical relation measured by the Euclidean distance between them. This is a simulation of the adjacency distance. Matching from the scene ARG to the model ARG is based on the unary type attribute and the binary distance relation. In the first experiment, a sub-part of 9 nodes in the dashed box of the model ARG is extracted as shown in Figure 9b in which the points re-labeled. This is to simulate an *exact* sub-ARG. The coordinates of the points of this sub-ARG are randomly deviated to generate an *inexact* version as in Figure 9c in which the points are re-labeled. Additional nodes reflecting noisy spots due to mal-segmentation are then added to this ARG to further corrupt the ARG as shown in Figure 9d in which the points are re-labeled and nodes 1, 2, 8 and 12 are the added. This is used as the simulated scene ARG derived from the segmentation of an image containing containing a visible part of the model object.

Table 2 illustrates the process of the labeling. Displayed at the first row of the table is the index i of the scene nodes. Displayed below the scene nodes are the model nodes S_i to which corresponding scene nodes are maximally associated at time t, *i.e.* S_i is chosen by $f^{(t)}(i, S_i) = \max_I f^{(t)}(i, I)$. The algorithm converges after about 20 iterations. An explanation of the rows with the superscripts "*" and "**" will be give shortly. Note that nodes 2 and 4 of the scene ARG are matched to node 5 of the model ARG. This is because they have the same node type and are close to each other. Therefore they are considered to have been split from model node (patch) 5 by the segmentation process. This is also, of course, the result due to that their relations to the rest of the nodes in their graphs are similar. Node 12 is matched to node 19 because indeed this can be the case. Nodes 1 and 8 are matched to the $NULL$ since there are no likely counterparts for them in the model ARG.

It is interesting to note that during iteration $t = 1, ..., 6$, all the scene nodes are most strongly associated with 0, *i.e.* they are the most strongly classified as the $NULL$ in the model ARG. After this period of "confusion", the correct interpretation gradually becomes more and more evident. This also implies that the correct solutions are achieved "via" the $NULL$ matches.

In the second experiment, a new scene is created from the model as shown in Figure 9e. It is done by firstly, removing some nodes, most of which are in the middle of the model, to generate a sub-ARG, secondly, randomly deviating the positions of the nodes of the sub-ARG, and then adding additional nodes to the deviated sub-ARG. This results the new scene. The matching result is shown in Table 3.

In Tables 2 and 3, the labeling process without the superscripts "*" and "**" are performed with $\pi_{2,0} = e^{-1}$ (see Eq. (15)). Shown in the rows with the superscripts are the final results produced with $\pi_{2,0} = e^{-2}$ and $\pi_{2,0} = e^{-0.5}$, respectively. The results demonstrates that $\pi_{2,0}$ has a significant yet robust control over the degree to which nodes are assigned 0 ($NULL$) due to inexactness. The larger $\pi_{2,0}$ is, the more nodes tend to be classified as the unknown.

These two experiments demonstrate the effectiveness of the proposed approach for inexact sub-ARG matching. The matching is independent of translation and 2D rotation since only the relative relations such as distance between nodes, but not absolute features such as the position of nodes and the orientation of graphs, are used. Whether the proposed ARG matching will perform as well for 3D surface matching depends on how well invariant descriptions of 3D surfaces can be extracted. If the scene ARG extracted from the segmentation of an image of a view of an object is to some extend 3D invariant, then it should approximate a sub-part of the object model ARG, and thus the proposed matching approach should work. To test this, we will present in Section 6 two experiments on matching between a Renault part model image and images viewed from different coordinates from the model.

In object recognition, there are generally more than one models. In this situation, the scene ARG is matched against each of model ARGs and the best-matched model is selected as the result of recognition. Let E_Λ by the remaining minimum energy given model Λ. Model $\Lambda^\dagger$ is selected if

$$E_{\Lambda^\dagger} < E_\Lambda, \qquad \forall \Lambda \neq \Lambda^\dagger$$

The pairing between the scene patches $i \in \{1, ..., m\}$ and the best-matched model patches $I \in \{0, ..., M^\dagger\}$ is contained in $\{f^*(i, I)\}$ found with the best-matched model $\Lambda^\dagger$. Recently, we have developed a connectionist approach based on the proposed framework in which segregation and inexact matching of overlapping ARGs are performed simultaneously [64].

5 Neural Networks

The computational algorithms at all the three levels are inherently parallel and distributed and can be efficiently implementing on parallel distributed architectures like the Connectionist Machine [41]. The dynamic systems we have constructed for solving the minimization problems at the low and the intermediate levels are equivalent to, or an extension of, those of the Hopfield neural networks [44, 45]. More exactly, the network for our low level computation belongs to the category in [45], and that for the intermediate level is an extension to [44]. In this section, we describe the neural networks for solving our problems, and compare them to those of Hopfield's. We also discuss a possibility of using the graded Hopfield model to perform the constrained minimization, or continuous relaxation labeling, at the high level.

5.1 Low Level

The dynamics of the low level process is determined by Eq. (16) rewritten below as

$$-\frac{1}{2\mu}\frac{df(i)}{dt} = [f(i) - d(i)] + 2\lambda \sum_{j \in \omega(i)} \pi(i,j)[f(i) - f(j)] \tag{19}$$

By letting $C = 1/2\mu$, $T(i,j) = 2\lambda\pi(i,j)$ and $R = 1$ and noting that $\omega(i) = \{i-1, i+1\}$ is the set of the direct adjacent points to i, we can write the above as

$$C\frac{df(i)}{dt} = -\frac{1}{R}f(i) + d(i) + T(i-1,i)[f(i-1) - f(i)] + T(i+1,i)[f(i+1) - f(i)]$$

According to Kirchoff current law, the above can be interpreted as the dynamic equation at neuron i of a massive analog neural network. Figure 10 shows the structure of this network, where $d(i)$ is the external input current source, $f(i)$ is the potential of neural cell i, $T(i,j)$ is the conductance or synaptic efficacy between neurons i and j, C is the membrane capacitance and R transmembrane resistance. The electrical current from i to j is $T(i,j)f(i,j)$. Discussions about analog network for regularization can be found in [13, 38, 58, 69, 79, 89, 98]. A recent development on adaptive regularization networks is reported in [63].

Eq. (19) can be compared directly to the Hopfield dynamics [45]. Let us re-arranged the equation

$$\frac{1}{2\mu}\frac{df(i)}{dt} = d(i) + 2\lambda \sum_{j\in\omega(i)} \pi(i,j)f(j) - [1 + 2\lambda \sum_{j\in\omega(i)} \pi(i,j)]f(i)$$

The Hopfield dynamics [45] is described by

$$C(i)\frac{du(i)}{dt} = I(i) + \sum_j T(i,j)V(j) - \frac{1}{R(i)}g^{-1}(V(i))$$

where $V(i) = g(u(i))$ and $g(\cdot)$ is a monotonically increasing function. The parallel between these two is obvious if we let in the latter $C(i) = 1/2\mu$, $I(i) = d(i)$, $T(i,j) = 2\lambda\pi(i,j)$, $R(i) = [1 + 2\lambda\sum_{j\in\omega(i)}\pi(i,j)]^{-1}$ and $g^{-1}(V) = V$.

Figure 10: ===== about here =====

5.2 Intermediate and High Levels

The problems in these two levels have basically the same nature: matching or labeling. Denote a set of labels by $\mathbf{D}_I$, $\mathbf{D}_I = \{1,2,3\}$ for the intermediate level or $\mathbf{D}_I = \{0,1,...,M\}$ for the high level. The goal is to find a mapping from $\{1,...,m\}$, which can be the set of pixels or scene ARG nodes, to $\mathbf{D}_I$ such that the energy is minimized. The final result is required to be consistent and unambiguous.

We have discussed two relaxation labeling algorithms for this tasks, the discrete one at the intermediate level and the continuous one at the high level. Features common to both algorithms are summarized as follows: The computation is done on a domain labeling $\mathbf{D}$. The gradient $q^{(t)}(i,I)$ is computed in parallel for all $(i,I) \in \mathbf{D}$. The state f is updated to reduce the energy. The updating is performed in parallel for all i, but needs to be normalized in the I direction to satisfy the consistency constraint. The fundamental differences between the two approaches lie in the spaces through which solutions are achieved, and the updating rules. The discrete rule is used at the intermediate level since it's cheaper yet satisfactory. The continuous rule of Hummel and Zucker [49] is used at the high level because it has a more favorable behavior. In this latter case, the updating is performed in a more careful and gradual way, and generally yields better solution.

Figure 11: ===== about here =====

The structure of the labeling network is illustrated in Figure 11. In the vertical direction is the scene index (i) whereas in the horizontal direction is the model index (I). There are $m \times \#\mathbf{D}_I$ nodes in the network. The state $f(i, I)$ reflects the strength with which i is associated to, or labeled, I. Note however that at the high level, some nodes may never be active since the unary symbolic constraints have singled out these impossible matches. The external input to node (i, I) of the network is determined by $d(i, I)$ while the connection between nodes (i, I) and (j, J) is determined by $\pi(i, I; j, J)$. There is a processor for each (i, I) which computes the gradient quantity $q(i, I)$. There is a processor for each i (shown in the figure as a horizontal line) which supervises the updating of labeling state based on $q(i, I)$ and $f(i, I)$, $\forall I$. The supervision determines the direction and the amount by which $f(i) = \{f(i, I) \mid \forall I\}$ is updated such that the energy is lowered while the consistency constraint is kept satisfied. In the following, we discuss the discrete and the continuous updating schemes of the labeling networks, and compare them to Hopfields'.

In the discrete updating at the intermediate level, the label assigned to i is updated according to the rule described in Section 4.2

$$\begin{gathered} f^{(t+1)}(i, S) \leftarrow 1 \\ f^{(t+1)}(i, R) \leftarrow 0 \\ \text{if} \\ q^{(t)}(i, S) < q^{(t)}(i, R) \qquad \forall R \neq S \end{gathered}$$

where

$$q^{(t)}(i, I) = -d(i, I) - 2\lambda \sum_{(j,J) \in \omega(i,I)} \pi(i, I; j, J) f^{(t)}(j, J)$$

This rule selects and activates the node at which the gradient is the minimal and suppresses the rest.

The above updating rule is an extension to the two-state Hopfield model [44]. The latter follows the following rule

$$V(i) \leftarrow \begin{cases} 0 & \text{if } I(i) + \sum_{i \neq j} T(i, j) V(j) < 0 \\ 1 & \text{if } I(i) + \sum_{i \neq j} T(i, j) V(j) > 0 \end{cases} \tag{20}$$

where $V(i) \in \{0, 1\}$ is the two-state output of node i. Now let us represent the two state neuron $V(i) \in \{0, 1\}$ by $f(i, I)$, $I \in \{0, 1\}$, with $f(i, 0) = 1$ corresponding to $V(i) = 0$ and $f(i, 1) = 1$ to $V(i) = 1$. Here f is subject to the unambiguity constraint $f(i, I) \in \{0, 1\}$ and the consistency constraint $\sum_I f(i, I) \equiv 1$. Define

$$T(i, I; j, J) = \begin{cases} T(i, j) & \text{if } I = J, i \neq j \\ 0 & \text{otherwise} \end{cases}$$

With this notation, the updating rule can be written as

$$\begin{gathered} f(i,S) \leftarrow 1 \\ f(i,R) \leftarrow 0 \\ \text{if} \\ q(i,S) < q(i,R) \qquad R \neq S \end{gathered}$$

where

$$q(i,I) = -d(i,I) - \sum_{(j,J)} T(i,I;j,J) f(j,J)$$

This is the updating rule we use at the intermediate level. The main difference between the Hopfield model and ours is that the Hopfield network performs a two category classification whereas ours can potentially handle any number of label categories.

In updating at the high level, the labeling state at a node is varied continuously in a projected direction while kept within the space $\mathbf{S}^+$ until convergence. A hardware implementation for this is not very attractive since the computation for gradient projection is not simple. However, the computation is parallel and distributed for all i, and could be efficiently implementation on such architecture as the Connectionist Machine. See [70] for a discussion on parallel distributed implementation of the Hummel-Zucker algorithm in multiprocessor architecture environment.

This latter constrained optimization could also be performed using the graded Hopfield model. There is a term in the Hopfield model, which we shall denote by $E^a(f)$, which can impose the unambiguity constraint

$$E^a(f) = \sum_{(i,I)} \int_0^{f(i,I)} \theta^{-1}(f) df$$

In the above, $f(i,I) \in [0,1]$ is the output or state of neuron (i,I), which is related to an *internal variable* $u(i,I)$ by

$$f(i,I) = \theta(u(i,I))$$

where $u(i,I) \in \mathbf{R}$ is an unbounded internal variable and $\theta(x)$ is usually a sigmoid function

$$\theta(u) = 1/[1 + e^{-u/T}]$$

controlled by a parameter T. In very high gain when $T \rightarrow 0^+$, $f(i,I)$ is forced 0 or 1 depending on whether $u(i,I)$ is positive or negative [45]. This means this term in effect imposes the unambiguity constraint when $T \rightarrow 0^+$. Using the method of Lagrangian multipliers, We can also encode the consistency constraint into the energy function. This is done by adding to it a consistency term

$$E^b(f) = \sum_i [\sum_I f(i,I) - 1]^2 = 0$$

This term has its minimum value of zero when the consistency constraint in Eq. (9) is satisfied. Now the constrained optimization minimizes the following functional

$$E'(f) = E^c(f) + \lambda E^s(f) + aE^a(f) + bE^b(f) \tag{21}$$

where a and b are weights. The dynamics for minimizing the four term energy function is characterized by

$$Cdu(i,I)/dt = d(i,I) + \lambda \sum_{(j,J)\in\omega(i,I)} \pi(i,I;j,J)f(j,J) - a\ u(i,I) - b[\sum_J f(i,J) - 1] \tag{22}$$

The capacitance C controls the convergence of the network. This updating rule will lead to a solution which is unambiguous and consistent if the parameters T, a and b are chosen correctly. Obviously, minimizing $E'(f)$ will lead to different optima from that specified in the original problem definition. However it is possible to achieve near-optimal solutions the original problem with the proper treatment for saddle points [78].

6 Experiments

We have presented and discussed some results on curvature-based segmentation and ARG matching previously. In this section, we demonstrate two results on 3D surface matching, illustrate parameters involved in the experiments and the computational times.

Figure 12: ===== about here =====

These two experiments are aimed at testing our algorithm for the curvature based 3D surface matching under rotation and occlusion. We are interested in non-planar rotations (in contrast, by planar rotation, we mean rotation in the $X - Y$ plane corresponding to the page) since obviously our algorithm is largely independent of planar rotations. To do this, we select a Renault part image as the "ground truth" model and create two images viewed from different positions from that the model is viewed from. Our method is illustrated as follows.

The "ground-truth" model is shown at the top of Figure 12. Shown from the left to the right of the figure are the range data (shaded), HK map and an illustrative ARG extracted from the map. We create two rotated and occluded scenes shown at the middle and the bottom rows from the model image. We shall refer to them as Scene-1 (middle row) and Scene-2 (bottom row), respectively. The Scene-1 image on the middle-left is created by rotating the model image by 30 degrees about the Y axis (lying *in* the page, pointing away from readers) while the Scene-2 image on the bottom-left by 45 degrees. After the rotations, some of the surface points are missing. The missing data are interpolated using some simple pre-processing techniques. Each scene is then segmented into the HK map in the middle column. Major curvature-sign patches have been extracted in the segmentation despite the rotations with missing data. The corresponding scene ARG is extracted from the HK map and other sources of information such as the original depth map, the depth discontinuity map and the first derivative maps. Recall that each node in an ARG is attributed by its HK label and 3D patch area, and between each pair of nodes is a relation measured by the minimum distance between corresponding patches. Tables 4, 5 and 6 show the HK label and the area of the model ARG and the scene ARGs.

The final matching results are shown in Tables 7 and 8. In both tables, the columns starting with "Model" are results produced with $\pi_{2,0} = e^{-1}$, those with "Model*" are results produced

Table 4: ===== about here =====

Table 5: ===== about here =====

$\pi_{2,0} = e^{-2}$, and those with "Model**" $\pi_{2,0} = e^{-0.5}$; more scene nodes are assigned 0 with higher $\pi_{2,0}$. The matching converges after 12 iterations for all the cases. Most of the patches are correctly matched as can be seen. Here we discuss mis-matches. There are two mistakes in matching from Scene-1 to the model: $2 \to 5$ and $21 \to 16$. The right answer should be $2 \to 2$ and $21 \to 28$. The reason for the first mistake is explained as follows. Scene-1 node 21 has to be close to Scene-1 nodes 14 and 18 to be matched to model node 28. However there is little support for this. The following favors the mis-match: (1) it is close to Scene-1 node 13, and (2) the distance $dist(21, 13)$ between Scene-1 node 21 and 13 is nearly the same as that $DIST(16, 21)$ between Model nodes 16 and 21 such that the compatibility $\pi(21, 16; 13, 21)$ is large. These local situations dominate the global optimization, resulting in the mistake. There are similar reasons for the other mis-match that Scene-1 node 2 is matched to Model node 4 instead of model node 2. In the model, Model node 2 is directly adjacent to Model nodes 13, 17, *etc.* However in the scene, Scene-1 node 2 is far from Scene-1 nodes 13 and 14; instead it is close to Scene-1 nodes 3, 8 and 10. This results in the second mis-match. Besides these two mis-matches, there are a suspicious match, $11 \to 14$, and a $NULL$ match, $17 \to 0$ in this experiment.

In the other experiment in matching Scene-2 to the model, there are a few erroneous and suspicious matches: $3 \to 5$, $18 \to 23$, $20 \to 31$ and $21 \to 32$. Similar reasons can be given for these mis-matches. But in contrast to the previous experiment, most of these mis-matches are due to errors in HK labels since we have not take HK label errors into account of the inexact matching. Nonetheless, the HK mistakes and mal-segmentation in both cases are largely due to the *artificial* rotations which have brought about severe distortions of the surfaces. Noise may not be so severe with commercially available dense range scanners of average quality. On the other hand, we have developed some post-verification algorithms to detect and correct the mis-matches (unpublished technical reports). This is not difficult to do using more global constraints since majority scene patches have been correctly matched. In summary of these results, the matching itself is successful since only two unary properties and one binary relation are needed despite a large degree of distortion is introduced in the images.

There are primarily seven parameters involved in the computation: (1) depth discontinuity threshold α in Eq. (8), (2) energy weight λ in Eq. (5) (3) time step constant μ in Eq. (18), (4) neighbor size r in defining ω in Eq. (6), (11) and (13), (5) curvature sign threshold τ in Eq. (10), (6) the control parameters σ_2^k in Eq. (7), (12) and (14), and (7) the compatibility $\pi_{2,0}$ for $NULL$ matches in Eq. (15). Table 9 illustrates the selection of these parameters, where

$$V = (1/m^2) \sum_{i,j=1}^{m} \left(|d(i+1,j) - 2d(i,j) + d(i-1,j)| + |d(i,j+1) - 2d(i,j) + d(i,j-1)| \right)$$

is a quantity which roughly measures the noise level of the input range data d. Table 10 gives an

Table 6: ===== about here =====

Table 7: ===== about here =====

impression on the average computational time for the experiments presented in this paper. The numbers listed are the sum of user time and system time on a SUN-4 Sparc station in seconds.

7 Related Work

Our review is divided into four topics: Optimization and networks, curvature based segmentation, relational matching and weak constraint satisfaction.

7.1 Optimization and Networks in Computer Vision

Many computer vision problems concern finding the most plausible interpretation of an image in terms of some criterion. It is natural that they are formulated in the framework of optimization. Optimization has found extensive fields at different levels of computer vision. Lower level examples are surface reconstruction,optical flow, edge detection, *etc.* [33, 47, 80]. These problems are solved in terms of variational principles [21]. Optimal criteria have also been developed for other aspects, *e.g.* image restoration, perceptual grouping. At higher level, optimization has been used for solving problems such as matching. Work in this regard will be discussed in more detail in Section 7.3.

Optimal solutions are usually found using some sort of search techniques since in most cases closed-form analytical solutions are unlikely. However, algorithms for searching optimal solutions differ. In this work we use gradient-guided search methods. A problem in gradient search is to avoid being trapped in a local minimum. Annealing methods are often used deal with this problem. There are basically two categories of annealing, namely, deterministic annealing such as graduated non-convexity (GNC) [13], and scale-space solution tracking [94], and stochastic annealing such as simulated annealing or Boltzmann machines [1, 31, 54]. In both approaches, the process is controlled by decreasing a parameter in the energy function which has an overall influence on system behavior. Examples of this parameter are the non-convexity control in GNC, the scale parameter in scale-space tracking and the temperature in simulated annealing. See [12] for a comparison between the two approaches.

We have discussed some relationships between the network models we used the Hopfield networks [44, 45] in the previous section. In fact, the gradient based optimization algorithms work through local propagation in a parallel and distributed way and therefore are suited for implementation on neural network architectures. Much research has been devoted to developing such architectures for solving lower level problems [13, 39, 48, 58, 69, 79, 89, 98], and higher level ones mostly of combinatorial optimization [46, 77, 99].

Table 8: ===== about here =====

Table 9: ===== about here =====

Table 10: ===== about here =====

7.2 Curvature Image Estimation and Segmentation

Currently, the most popularly adopted methodology for the reliable estimation of derivatives is local surface fitting. Refinement of segmented maps is usually achieved using region growing and split-and-merge. In this regard, the most extensive work is probably that by Besl and Jain [5, 6, 7] Derivatives extracted by surface fitting are used for computing the surface curvature. Low order surface models are fitted after the depth data have been convolved with window operators which combine smoothing and differentiation. The coarse segmentation is refined by region growing.

Similar work has been done earlier by Haralick, Watson and Laffy [37]. They classify surfaces into the topographic types defined in terms of the principle curvature. A local *facet model* is fitted to the data to estimate partial derivatives. From this, directional derivatives and thereby principal curvature are computed analytically.

In other related segmentation work, Faugeras and Hebert [27] segment range data into planar or quadric patches by region growing. Hoffman and Jain [43] present a procedure for segmentation and classification of range images into planar, concave and convex patches. They apply statistical tests and region merging to estimate surface normals, after nearest-neighbor smoothing to reduce noise. Yokoya and Levine [97] estimate derivatives by fitting the range data to a local quadric surface model within a best-selected window which minimizes a fitting error. They refine the initial segmentation by contrasting followed by expanding. The surface segmentation is combined with jump and roof edge maps to form a "hybrid" segmentation result. Medioni and Nevatia [71] estimate zero-crossings and maxima of surface curvature properties by first smoothing data with a large window and then computing directional derivatives using 1D masks. Fan, Medioni and Nevatia [24] have improved on this previous work using two separate procedures: a procedure that estimates and tracks directional curvature under different scales and a simpler one for less noisy images.

Continuity-controlled smoothing problem is intensively studied in the context of surface reconstruction [13, 14, 34, 58, 69, 79, 89]. In the context of computing curvature properties, Brady *et al.* [19] use the technique of computational molecules [88] for discontinuity-controlled smoothing. They perform repeated averaging smoothing using the molecules to prevent over-smoothing across discontinuities. Li [61], inspired by [59], attempts to avoid smoothing across discontinuities by decomposing the profiles of an image into piecewise curve segments at discontinuities before performing derivative estimation using a surface fitting model.

7.3 Attributed Relational Graph Matching

Literature on this is abundant. In what follows, only part of the work relevant to inexact matching, matching by relaxation, and relational graph matching is reviewed and compared with the work presented here. The work on matching by explicit searching, maximal cliques and others, although important, are not discussed further.

Shapiro and Haralick [86] propose a method for inexact matching. They study a class of inexact constraint satisfaction problems, which is a more relaxed, or in other words, less constrained, version of their previous work on consistent labeling [36]. An *ϵ-homomorphism* is defined as a mapping in which the weighted number of mis-mapped *symbolic* relations is less than the threshold ϵ. Errors and mistakes in symbolic relations are mapped into a number, and this number is then used as a criterion to decide whether a homomorphism is acceptable or not. The inexact matching problem is viewed as finding the best homomorphism, i.e. the one with minimum number of errors with respect to a given attribute value threshold, a missing part threshold, and a relation threshold. The ϵ-homomorphisms are found by *look-ahead* search techniques. The scheme is used to do object matching where object models are composed of object parts and relations among the parts (see also [87]).

Error measures are also proposed from other view points. For example, Kitchen [55] proposes a criterion called the *goodness of fit* function, as a measure of quantitative relational distances. Here two feature vectors of quantitative *unary* relations are mapped into a number in the range [0,1]. Sanfeliu and Fu [84] and Eshera and Fu [23] define the distance between two relational structures as the minimum number of modifications needed to transform one into the other.

Davis [22] uses relaxation techniques to match two 2D polygonal shapes. His work shares many similarities with ours. It is formulated in the optimization framework. Real-valued constraints are used to define a compatibility function and a relaxation algorithm is then used for the optimization. Corner points are the entities to be matched. Corner angles and the distances between corners are used as the attribute and the relation for matching.

His optimization criterion consists of four terms. The first term measures the difference between angles. The second term measures the difference of the tension in the springs connecting corners in the model and those in the scene. The final two terms penalize descriptions that are not matched. The idea of the above criterion dates back to the work by Fischler and Elschlager [28].

Davis also uses an association graph for matching relational structures. Nodes in this graph are connected if the compatibility is greater than a threshold. The four-term criterion is optimized by using search procedures. Ambiguity is reduced by a node-deletion process in which nodes are deleted from the graph if some evaluation function is below a threshold. This process generates a sequence of association graphs until the fixed point is reached. The concept of association graphs was first introduced by Ambler *et al.* [2] and forms the basis of a class of labeling networks.

The work by Cheng and Huang [20] and that of Radig [81] aims at finding correspondence between two images using relational matching. Chen and Huang's approach to image registration can deal with many-to-one matching. Each of the four binary relations used, namely parallelism, antiparallelism, collinearity and adjacency, are associated with numerical attributes. In the work on sequence analysis by Radig, a paradigm is proposed for finding correspondence between images and prototypes. This is done by searching for maximal cliques. The issue of many-to-one mappings is addressed. Symbolic relations are attached attribute values. Whether or not a match is accepted depends on whether or not the difference between a pair of attributed relations is smaller than a threshold.

In the work by Bhanu [8], matching 3D planar surface shape is posed as an optimization problem. Criteria of labeling consistency are minimized by stochastic relaxation labeling [26]. Compatibility information is determined according to the average error computed from transformations found between a view and the model. Object surfaces are segmented into 3D planar faces but no explicit binary relations are used.

In a range-based 3D vision system developed by Oshima and Shirai [75], range images are segmented into planar and smoothly curved surface patches. A relational-feature graph is constructed to represent the segmented patches and their relations. Data-driven and model-driven searching is performed for matching between the scene and models. Large planar patches with no occlusion are extracted as kernels. These kernels are used to find probable models containing the kernel patches. Correspondences to other scene patched are found based on matches between the kernel patches and model patches.

Fan *et al.* [25] and Jain and Hoffman [50] use representations derived from curvature-based surface segmentation for 3D object recognition from range images. In the work by Fan *et al.* , object surface segmentation is based on *zero-crossings* and *extrema* of surface curvature. The regions surrounded by the resulting boundaries from these curvature points are assumed to correspond to elementary surface patches. The attributes of patches and the adjacency relations between them are measured and extracted, and are organized into a representation called the *Attributed Graph (AG).* Their AGs consist of nodes (surface patches) and adjacency relations between the nodes. The attributes of nodes include the area of a patch, its orientation, the direction of the normal or least curvature, the average principle curvature, an estimation of ratio of occlusion, and the 3D centroid coordinates. The adjacency relations are characterized by their *symbolic* types such as jump, limb, convex and concave crease. A scene AG and a model AG are matched using tree search techniques and the transformation between the scene and the model are computed. The initial results are verified and modified by an analyzer. The results of matching are used in decision making for object recognition.

Jain and Hoffman adopt a coarse representation for surfaces based on differential geometry. They classify surfaces as being planar, convex and concave to obtain a more stable surface decomposition. They also detect edges of jump or crease type. A rule-based system matches a scene against a model using a Bayesian reasoning and inference strategy.

Wong *et al.* [95] present a vision system for 3D shape synthesis and recognition based on an *attributed hypergraph representation (AHR).* An AHR is a hierarchical structural representation of objects. It consists of (1) the elementary area attribute graph for each surface, (2) the primitive block attribute for each component block, and (3) the AHR for objects(s). The AHR suggests a method of combining a representation based on the invariant properties, and one based on decomposition by parts. Model AHRs are constructed either by directly measuring models or by synthesizing individual AHRs from multiple views. The matching between two AHRs is monomorphic, *i.e.* of one-to-one.

Yang *et al.* [96] use graph matching for scene labeling. Their definition of compatibility function is based on symbolic relations. However it has directly inspired our definition based on numerical relations and our design of the labeling networks. In their work, objects are described by a graph consisting of adjacency relations and unary properties of surfaces. An associated graph is formed from the scene graph and a model graph in a similar way to [2] and the matching is then performed by finding the maximal cliques [2, 15, 32].

Recently, neural networks have been used for image matching and object recognition. Jamison and Schalkoff [52] use a Hopfield-like network for image labeling where a scene is described by symbols. They relate neural networks to the labeling problem and formulated an energy function for constraint satisfaction. Nasrabadi *et al.* [73] use a Hopfield network to perform subgraph isomorphism for matching overlapping 2D object. Symbolic feature compatibility is used to compute network connections.

7.4 Weak Constraint Satisfaction

The inexact matching described in this work share some aspects with *weak constraint satisfaction.* In weak constraint satisfaction paradigm, "hard" constraints are allowed to be violated without causing the failure of constraint satisfaction. However, each such violation is penalized by adding an amount to a cost function that measures the imperfectness of the matching. Blake and Zisserman [11, 13], Koch *et al.* [58] and Terzopoulos [89] use weak membrane and/or weak plates for surface reconstruction from noisy data allowing for discontinuities. Marroquin [69] solves a similar problem, but instead of using deterministic gradient descent in minimizing the associated cost energy functions, he uses the stochastic approach of Geman and Geman [31]. In a *weak notion of graph-matching*, Bienenstock [10] proposes a scheme for an approximation of graph isomorphism where relation-preserving characteristics of isomorphism can be violated. However, each violation incurs a small penalty, in the spirit of weak constraint satisfaction. Nevertheless, at a higher level where more abstract representations are used, the weak constraint satisfaction problem becomes more complicated. For example in our application where matching has to be homomorphic or even less constrained, it is unclear how to establish sensibly the weak notion from two ARG's using a term of penalty.

Recently, another class of regularizers termed adaptive regularizers (ARs) are proposed for continuity-controlled reconstruction in low level vision [63]. This is based on the idea of using "soft" constraints to avoid possible failure of tasks due to a few violations of "hard"constraints. In the ARs, the connection between neighboring points of the computational networks are defined by a real number between 0 and 1, rather than in $\{0, 1\}$ as *binary* line-process variables popularly used as in [13, 31, 58, 69, 89], *etc.* The continuous control of connections in ARs could be more preferable than the discrete control in the aforementioned work in that the continuous characteristics of the former is easier to be implemented using analog circuitry.

8 Conclusion

We have proposed a unified optimization framework to solve the low, intermediate and high level computer vision problems in 3D object recognition from range data. In this framework, solutions to the problems are found by minimizing energy functions consisting of two terms which encode, respectively, unary and binary constraints. The minimization is performed using parallel and distributed relaxation algorithms suited for neural implementation.

We have presented experimental results to demonstrate the effectiveness and performance of our framework and algorithms. Segmentation results produced by the first two levels, we believe, have been a great improvement both in terms of the coherence of segmented surface patches and the preservation of surface shape near depth discontinuities. Considerable effort on the high level has gone into preventing the matching process from suffering from "hard" adjacency violation, and combining both symbolic and numerical constraints into the labeling networks. The results on the ARG matching and the 3D surface matching have demonstrated that the matching approach we have proposed is effective in handling relational violations. The approach is attractive in that it works in a parallel and distributed way. A recent work based on these matching ideas shows that the matching network can perform segregation and inexact matching of overlapping ARGs simultaneously [64].

The definitions of the connections of our computational networks have played important roles in accomplishing the tasks. At the first two levels, the connections define the interaction between neighbors of the networks. They have effectively prevented smoothing across depth discontinuities. At the high level, our definition of the connections has prevented inexact matching from possible failure in situations where "hard" constraints are violated. Furthermore, it has embodied both symbolic and numerical constraints into the computational networks in a unified way. Choosing to make each connection (or input) symbolic or numerical is controlled by a parameter. The inexact matching approach is significant not only in this application but also in general relational matching problems in computer vision.

We have offered insight into the unified framework by forging links, on one hand, between the segmentation at the first two levels and the matching at the high level, and on the other hand, between the regularization at the low level and the relaxation labeling the latter two levels. Seemingly very different, all the tasks perform a mapping from input to some models. The differences lie in the models and the set of labels mapped to. A model is either an explicit object (at the high level) or the implicit smoothness constraint in the physical world (at the first two levels). The set of labels is either finite (at the latter two levels) or infinite (at the low level). The inter-relationships between the low level regularization and higher level labeling have given an interpretation of the regularization techniques from a new perspective, that in terms of inexact relational matching.

Acknowledgment

Part of this work was done in Artificial Intelligence Department, Edinburgh University during author's first year Ph.D research under the supervision of Bob Fisher. Joseph Kittler, as the successive supervisor, provided an opportunity for this work to be continued in the Vision, Speech and Signal Processing (VSSP) Group, and gave helpful comments and suggestions for this research. My colleagues Ata Etemadi, Edwin Hancock, Graeme Jones, Maria Petrou commented on this and related manuscripts. Tony Carraro, Isaac Ng, Paolo Remagnino, Anthony Tai, K.C. Wong and Fan S. Wu offered their assistance in experiments. Anonymous referees' criticisms led to better presentation of the work. I am grateful to all of them. The work was supported by a scholarship from the British Council.

References

[1] D. H. Ackley, G. E. Hinton, and T. J. Sejnowski. "A learning algorithm for Boltzmann machines". *Cognitive Science*, 9:147–169, 1985.

[2] A. P. Ambler, H. G. Barrow, C. M. Brown, R. M. Burstall, and R. J. Popplestone. "A versatile computer-controlled assembly system". In *Proceedings of International Joint Conference on Artificial Intelligence*, pages 298–307, 1973.

[3] B. M. Bennett and D. D. Hoffman. "Shape decompositions for visual recognition: The role of transversality". In W. Richards, editor, *Image Understanding*, pages 215–256. Norwood, Ablex, NJ, 1987.

[4] M. Bertero, T. A. Poggio, and V. Torre. "Ill-posed problems in early vision". *Proceedings of the IEEE*, 76(8):869–889, August 1988.

[5] P. J. Besl. *Surfaces in Range Image Understanding.* Springer-Verlag, New York, 1988.

[6] P. J. Besl and R. C. Jain. "Three-Dimensional object recognition". *Computing Surveys*, 17(1):75–145, March 1985.

[7] P. J. Besl and R. C. Jain. "Segmentation through variable-order surface fitting". *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 9(2):167–192, 1988.

[8] B. Bhanu. "Representation and shape matching of 3D objects". *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6(3):340–350, May 1984.

[9] B. Bhanu. "CAD-based robot vision". *Computer*, 20(8):13–16, August 1987.

[10] E. Bienenstock. "Neural-like graph-matching techniques for image-processing". In D. Z. Anderson, editor, *Neural Information Processing Systems*, pages 211–235. Addison-Wesley, 1988.

[11] A. Blake. "The least disturbance principle and weak constraints". *Pattern Recognition Letters*, 1:393–399, 1983.

[12] A. Blake. "Comparison of the efficiency of deterministic and stochastic algorithms for visual reconstruction". *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(1):2–12, January 1989.

[13] A. Blake and A. Zisserman. *Visual Reconstruction.* MIT Press, Cambridge, MA, 1987.

[14] R. M. Bolle and B. C. Vemuri. "On three-dimensional surface reconstruction methods". *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(1):2–13, January 1991.

[15] R. C. Bolles. "Robust feature matching through maximal cliques". In *Proc. Soc. Photo-Opt. Intrum. Engs*, volume 182, pages 140–149, April 1979.

[16] R. C. Bolles and P. Horaud. "3DPO: A three-dimensional part orientation system". *International Journal of Robotic Research*, 5(3):3–26, Fall 1986.

[17] J. Brady, N. Nandhakumar, and J. Aggarwal. "Recent progress in the recognition of objects from range data". *Image and Vision Computing*, 7(4):295–307, 1989.

[18] M. Brady. "Criterion for describing surfaces". In J. Beck, B. Hope, and A. Rosenfeld, editors, *Human and Machine Vision.* Academic Press, 1983.

[19] M. Brady, J. Ponce, A. Yuille, and H. Asada. "Describing surfaces". *Computer Vision, Graphics and Image Processing*, 32:1–28, 1985.

[20] J. K. Cheng and T. S. Huang. "Image registration by matching relational structures". *Pattern Recognition*, 17(1):149–159, 1984.

[21] R. Courant and D. Hilbert. *Methods of Mathematical Physics*, volume 1. Interscience Publishers Inc., New York, 1953.

[22] L. S. Davis. "Shape matching using relaxation techniques". *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1(1):60–72, January 1979.

[23] M. A. Eshera and K. S. Fu. "A graph distance measure for image analysis". *IEEE Transactions on Systems, Man and Cybernetics*, SMC-14(3):398–408, May/June 1984.

[24] T. J. Fan, G. Medioni, and R. Nevatia. "Surface segmentation and description from curvature features". In *Proc. 1987 DARPA Image Understanding Workshop*, pages 351–359, Los Angeles, CA, February 1987.

[25] T. J. Fan, G. Medioni, and R. Nevatia. "Recognizing 3D objects using surface descriptions". *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(11):1140–1157, November 1989.

[26] O. D. Faugeras and M. Berthod. "Improving consistency and reducing ambiguity in stochastic labeling: An optimization approach". *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 3:412–423, April 1981.

[27] O. D. Faugeras and M. Hebert. "The representation, recognition and locating of 3D objects". *International Journal of Robotic Research*, 5(3):27–52, Fall 1986.

[28] M. Fischler and R. Elschlager. "The representation and matching of pictorial structures". *IEEE Transactions on Computers*, C-22:67–92, 1973.

[29] P. J. Flynn and A. K. Jain. "On reliable curvature estimation". In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 110–116, June 1989.

[30] P. J. Flynn and A. K. Jain. "CAD-based computer vision: From CAD models to relational graphs". *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(2), February 1991.

[31] S. Geman and D. Geman. "Stochastic relaxation, Gibbs distribution and the Bayesian restoration of images". *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6(6):721–741, November 1984.

[32] D. E. Ghahraman, A. K. C. Wong, and T. Au. "Graph optimal monomorphism algorithms". *IEEE Transactions on Systems, Man and Cybernetics*, SMC-10(4):181–188, April 1980.

[33] W. E. L. Grimson. *From Images to Surfaces: A Computational Study of the Human Early Visual System*. MIT Press, Cambridge, MA, 1981.

[34] W. E. L. Grimson and T. Pavlidis. "Discontinuity detection for visual surface reconstruction". *Computer Vision, Graphics and Image Processing*, 30:316–330, 1985.

[35] C. Hansen and T. C. Henderson. "CAGD-based computer vision". *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(11):1181–1193, 1989.

[36] R. M. Haralick and L. Shapiro. "The consistent labeling problem : part 1". *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1:129–139, April 1979.

[37] R. M. Haralick, L. Watson, and T. J. Laffey. "The topographic primal sketch". *International Journal of Robotic Research*, 2(1):50–72, Spring 1983.

[38] J. G. Harris. "A new approach to surface reconstruction: The coupled depth/slope model". In *Proceedings of First International Conference on Computer Vision*, pages 277–283, London, England, June 1987.

[39] J. G. Harris and A. M. Flynn. "Object recognition using the connection machine's router". In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Miami Beach, Florida, June 1986.

[40] E. C. Hildreth and S. Ullman. "The computational study of vision". *A. I. Lab. Memo* No. 1038, MIT, Cambridge, MA, November 1988.

[41] W. D. Hillis. *The Connection Machine.* MIT Press, Cambridge, MA, 1985.

[42] D. Hoffman and W. A. Richards. "Parts of recognition. *Cognition*, 18:65–96, 1985.

[43] R. L. Hoffman and A. K. Jain. "Segmentation and classification of range images". *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 9(5):608–620, September 1987.

[44] J. J. Hopfield. "Neural networks and physical systems with emergent collective computational abilities". *Proceedings of National Academic Science, USA*, 79:2554–2558, 1982.

[45] J. J. Hopfield. "Neurons with graded response have collective computational properties like those of two state neurons". *Proceedings of National Academic Science, USA*, 81:3088–3092, 1984.

[46] J. J. Hopfield and D. W. Tank. " 'Neural' computation of decisions optimization problems". *Biological Cybernetics*, 52:141–152, 1985.

[47] B. K. P. Horn. *Robot Vision.* MIT Press, Cambridge, MA, 1986.

[48] B. K. P. Horn. "Parallel networks for machine vision". *A. I. Memo.* No. 1071, MIT, December 1988.

[49] R. A. Hummel and S. W. Zucker. "On the foundations of relaxation labeling process". *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 5(3):267–286, May 1983.

[50] A. K. Jain and R. Hoffman. "Evidence-based recognition of 3D objects". *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 10(6):783–802, November 1988.

[51] R. C. Jain and A. K. Jain. "Report on Range Image Understanding Workshop, East Lansing, Michigan, March 21–23, 1988". *Machine Vision and Applications*, 2:45–60, 1989.

[52] T. A. Jamison and R. J. Schalkoff. "Image labeling: a neural network approach". *Image and Vision Computing*, 6(4):203–214, November 1988.

[53] R. A. Jarvis. "A perspective on range finding techniques for computer vision". *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 5(2):122–139, 1983.

[54] S. Kirkpatrick, C. D. Gellatt, and M. P. Vecchi. "Optimization by simulated annealing". *Science*, 220:671–680, 1983.

[55] L. Kitchen. "Relaxation applied to matching quantitative relational structures". *IEEE Transactions on Systems, Man and Cybernetics*, SMC-10(2):96–101, February 1980.

[56] J. Kittler and E. R. Hancock. "A contexture decision rule based on triples of object labels". *Alvey Vision Club Conference*, September 1986.

[57] J. Kittler and J. Illingworth. "Relaxation labeling algorithms – a review". *Image and Vision Computing*, 3(4):206–216, November 1985.

[58] C. Koch, J. Marroquin, and A. Yuille. "Analog 'neuronal' networks in early vision". *Proceedings of National Academic Science, USA*, 83:4263–4267, 1986.

[59] Y. G. Leclerc and S. W. Zucker. "The local structure of image discontinuities in one dimension". *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 9(3):341–355, May 1987.

[60] S. Z. Li. "3D object recognition from range images: Computational framework and neural networks". In *Proceedings of 2nd International Conference on Intelligent Autonomous Systems*, pages 100–110, Amsterdam, Netherlands, December 1989.

[61] S. Z. Li. "A curve analysis approach to surface feature extraction from range image". In *Proceedings of International Workshop on Machine Intelligence and Vision*, Tokyo, Japan, April 1989.

[62] S. Z. Li. "Invariant surface segmentation through energy minimization with discontinuities". *International Journal of Computer Vision*, 5(2):161–194, 1990.

[63] S. Z. Li. "Reconstruction without discontinuities". In *Proceedings of Third International Conference on Computer Vision*, pages 709–712, Osaka, Japan, December 1990.

[64] S. Z. Li. "Mapping from pixels to models: A connectionist approach". In *Proceedings of The 7th Scandinavian Conference on Image Ananlysis*, Copenhagen, Denmark, August 1991.

[65] M. Lipschutz. *Differential Geometry*. McGraw-Hill, 1969.

[66] S. C. Liu and J. G. Harris. "Generalized smoothing networks in early vision". In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 184–191, 1989.

[67] D. Marr. *Vision*. W. H. Freeman and Co, San Francisco, 1982.

[68] D. Marr and H. K. Nishihara. "Representation and recognition of the spatial organization of three dimensional structure". *Proceedings of Royal Society, London*, B200:269–294, 1977.

[69] J. L. Marroquin. "Probabilistic solution of inverse problems". *A. I. Lab. Tech. Report* No. 860, MIT, Cambridge, MA, 1985.

[70] B. M. McMillin and L. M. Ni. "A reliable parallel algorithm for relaxation labeling". In P. M. Dew, R. A. Earnshaw, and T. R. Heywood, editors, *Parallel Processing for Computer Vision and Display*, pages 190–209. Addison-Wesley, 1989.

[71] G. Medioni and R. Nevatia. "Description of 3D surfaces using curvature properties". In *Proc. 1984 DARPA Image Understanding Workshop*, pages 291–299, New Orleans, LA, October 1984.

[72] J. Mohammed, R. Hummel, and S. Zucker. "A feasible direction operator for relaxation method". *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 5(3):330–332, May 1983.

[73] N. Nasrabadi, W. Li, and C. Y. Choo. "Object recognition by a Hopfield neural network". In *Proceedings of Third International Conference on Computer Vision*, pages 325–328, Osaka, Japan, December 1990.

[74] D. Nitzan. "Three-dimensional vision structure for robot applications". *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 10(3):291–309, May 1988.

[75] M. Oshima and Y. Shirai. "Object recognition using three-dimensional information". *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 5(4):353–361, July 1983.

[76] A. P. Pentland. "Recognition by parts". In *Proceedings of First International Conference on Computer Vision*, London, England, June 1987.

[77] C. Peterson and B. Soderberg. "A new method for mapping optimization problems onto neural networks". *International Journal of Neural Systems*, 1(1):3–22, 1989.

[78] J. C. Platt and A. H. Barr. "Constrained differential optimization". In *Proceedings of the IEEE 1987 NIPS conference*, 1988.

[79] T. Poggio and C. Koch. "Ill-posed problems in early vision: from computational theory to analog networks". *Proceedings Royal Society London*, B(226):303–323, 1985.

[80] T. Poggio, V. Torre, and C. Koch. "Computational vision and regularization theory". *Nature*, 317:314–319, 1985.

[81] B. Radig. "Image sequence analysis using relational structures". *Pattern Recognition*, 17(1):161–167, 1984.

[82] F. Rosenblatt. *Principles of Neurodynamics.* Spartan, New York, 1962.

[83] A. Rosenfeld, R. Hummel, and S. Zucker. "Scene labeling by relaxation operations". *IEEE Transactions on Systems, Man and Cybernetics*, 6:420–433, June 1976.

[84] A. Sanfeliu and K. S. Fu. "A distance measure between attributed relational graphs for pattern analysis". *IEEE Transactions on Systems, Man and Cybernetics*, SMC-9:757–768, 1983.

[85] J. Serra. *Image Analysis and Mathematical Morphology.* Academic Press, 1982.

[86] L. G. Shapiro and R. M. Haralick. "Structural description and inexact matching". *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 3:504–519, September 1981.

[87] L. G. Shapiro, J. D. Moriarty, R. M. Haralick, and P. G. Mulgaonkar. "Matching three-dimensional objects using a relational paradigm". *Pattern Recognition*, 17(4):385–405, 1984.

[88] D. T. Terzopoulos. "The role of constraints and discontinuities in visible surface reconstruction". In *Proc. 8th International Joint Conference on AI*, pages 1073–1077, Karlsruhe, W. Germany, 1983.

[89] D. T. Terzopoulos. "Regularization of inverse visual problems involving discontinuities". *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 8(4):413–424, July 1986.

[90] A. N. Tikhonov and V. A. Arsenin. *Solutions of Ill-posed Problems.* Winston & Sons, Washington, 1977.

[91] S. Ullman. *The Interpolation of Visual Motion.* MIT Press, Cambridge, MA, 1979.

[92] S. Ullman. "Relaxation and constraint optimization by local process". *Computer Graphics and Image Processing*, 10:115–195, 1979.

[93] B. Vemuri and J. Aggarwal. "3D model construction from multiple views using range and intensity data". *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 435–437, June 1986.

[94] A. Witkin, D. T. Terzopoulos, and M. Kass. "Signal matching through scale space". *International Journal of Computer Vision*, 1:133–144, 1987.

[95] A. K. C. Wong, S. W. Lu, and M. Rioux. "Recognition and shape synthesis of 3D objects based on attributed hypergraph". *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(3):695–703, March 1989.

[96] B. Yang, W. E. Snyder, and G. L. Bilbro. "Matching oversegmented 3D images to models using associated graphs". *Image and Vision Computing*, 7(2):135–143, May 1989.

[97] N. Yokoya and M. D. Levine. "Range image segmentation based on differential geometry: A hybrid approach". *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(6):643–649, June 1989.

[98] A. L. Yuille. "Energy function for early vision and analog networks". *A. I. Lab. Memo* No. 987, MIT, Cambridge, MA, 1987.

[99] A. L. Yullie. "Generalized deformable models, statistical physics and matching problems". *Neural Computation*, 2:1–24, 1990.

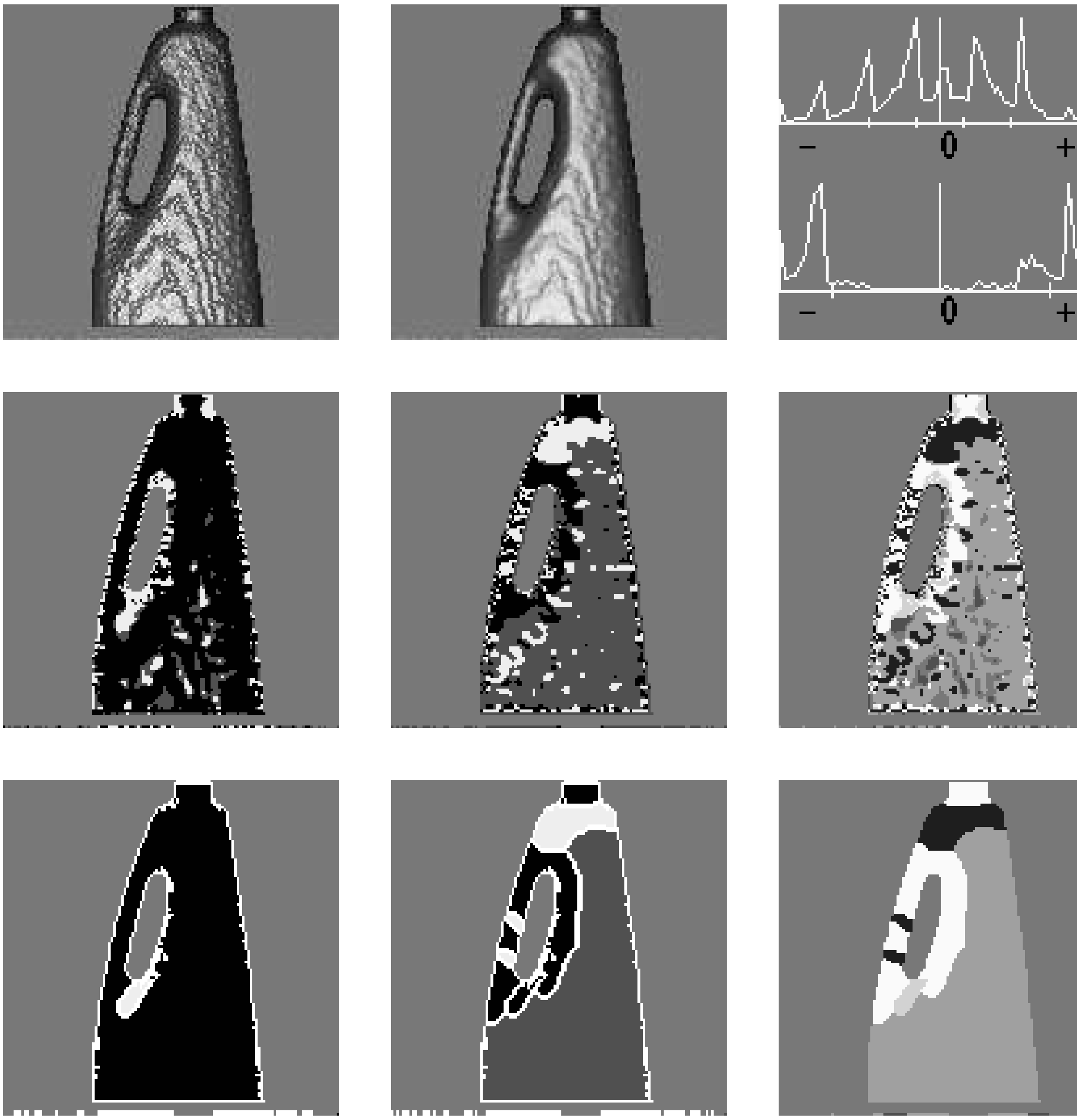


Figure 2: Segmentation of a bottle image. Top: original image, smoothed image and histograms of H and K images. Middle: H, K and HK maps obtained by thresholding. Bottom: refined H, K and HK maps.

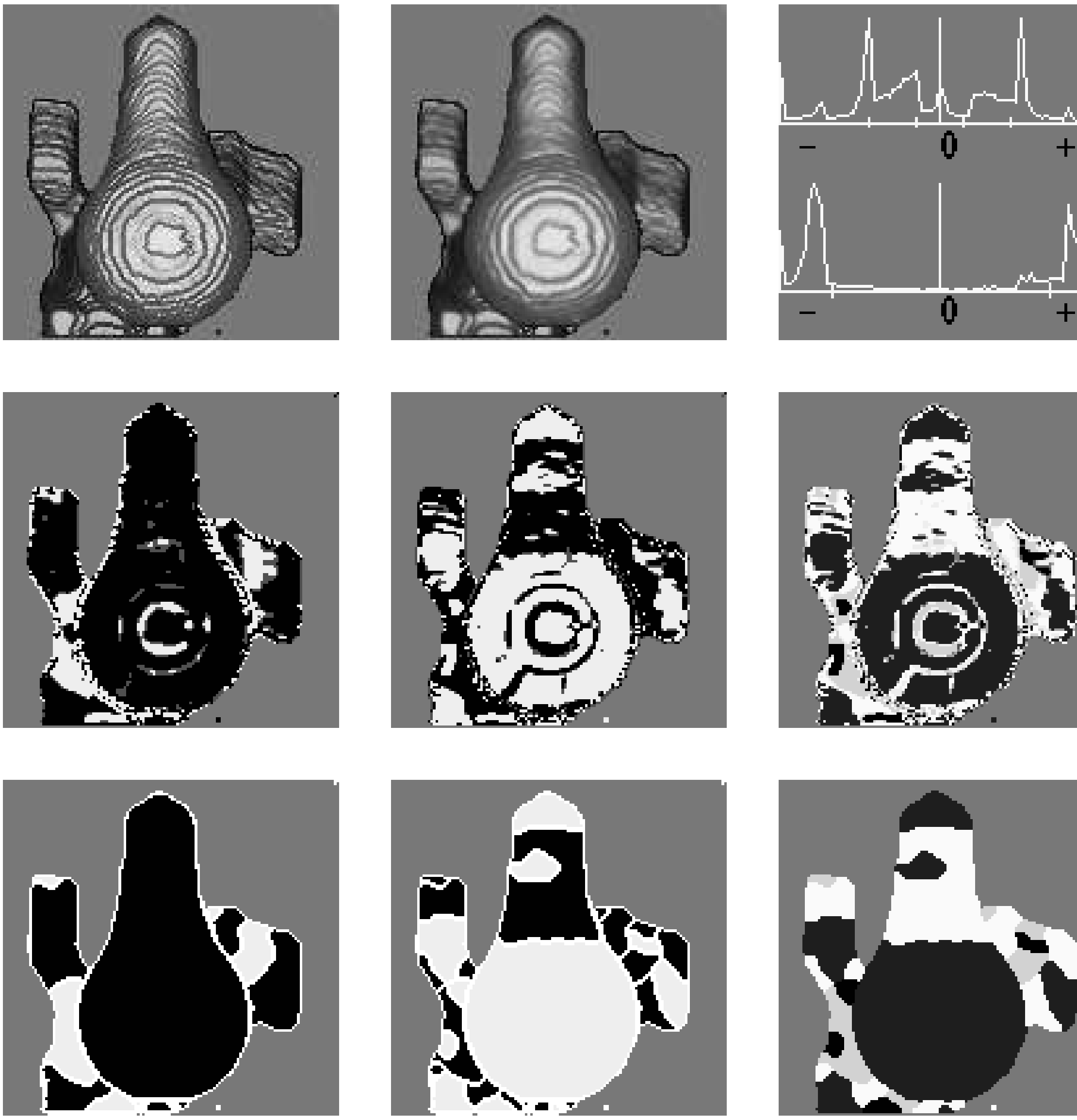


Figure 3: Segmentation of an image containing a Renault part superimposed by a light bulb. Top: original image, smoothed image and histograms of H and K images. Middle: H, K and HK maps obtained by thresholding. Bottom: refined H, K and HK maps.

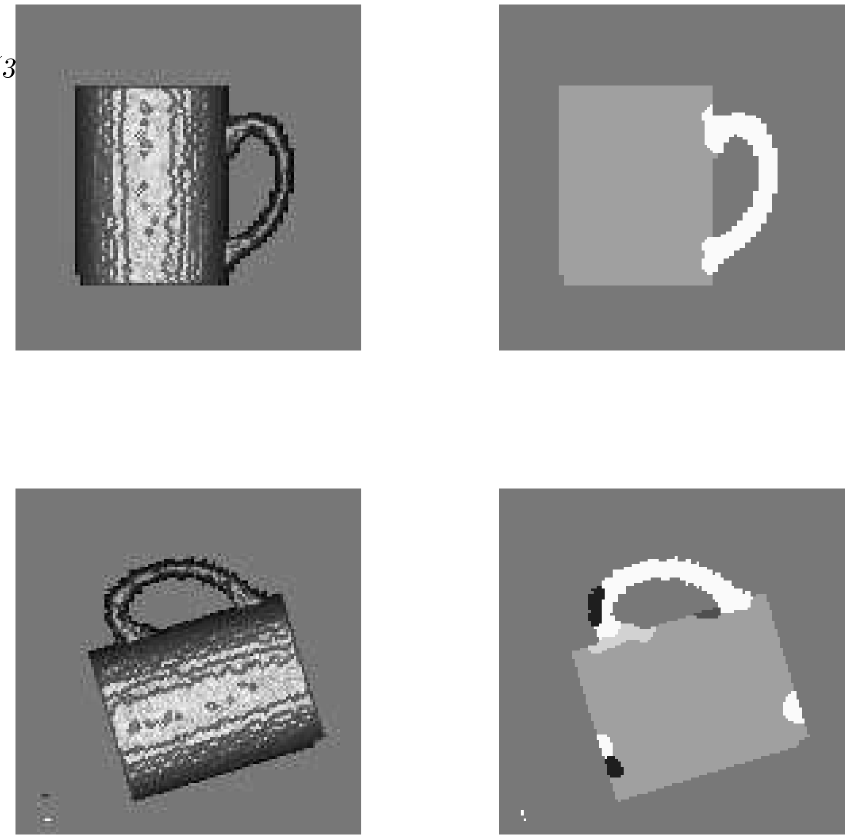
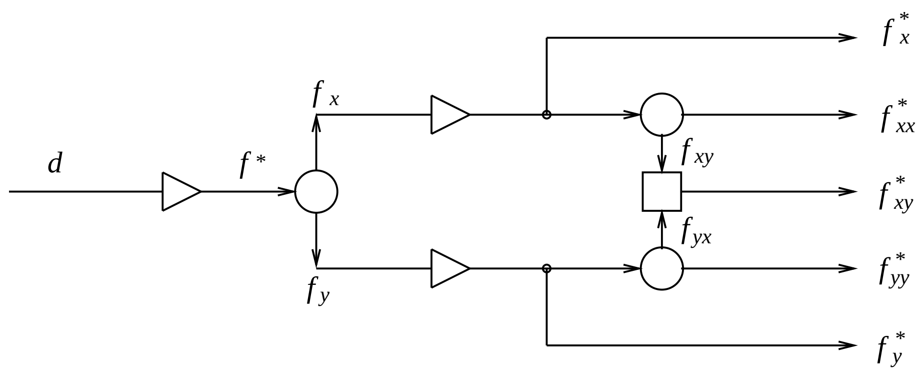


Figure 4: Cup images and their HK maps. Top: a cup image and its HK map. Bottom: a transformed version of the cup image and the HK map. Some noise is introduced during the transformation due to quantization.

Figure 5: ARGs built from the cup HK maps. (a) ARG from the top-right of FIG.3. Node 1 corresponds to the body of the cup and node 2 to the handle. (b) ARG from the bottom-right of FIG.3. Node 5 corresponds to the body, node 1 to the handle, and the rest to $NULL$. (c) Texture-HK label correspondence.

Figure 6: Diagram of the system structure.

Figure 7: An illustration of the partial derivative estimation process. A triangle represents a regularization operation, a circle a finite difference operation and the box the mean-averaging operation.



```
    o o o            o
    o o o        o o o o o
  o o x o o      o o x o o
    o o o        o o o o o
    o o o            o
```

Figure 8: Two candidate sets of the neighbors of point 'x'.

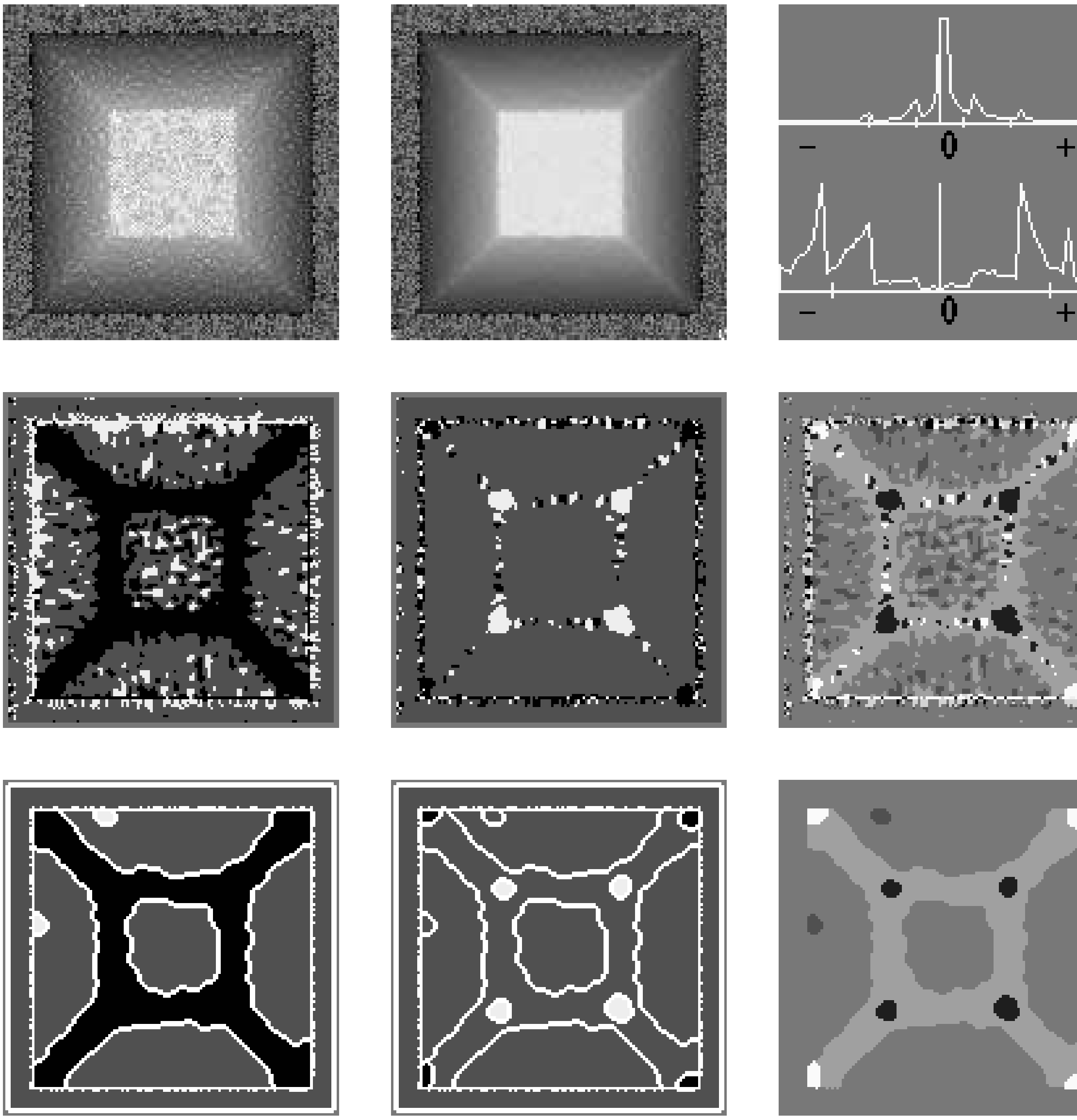


Figure 9: Segmentation of a pyramid image. Top: original image, smoothed image and histograms of H and K images. Middle: H, K and HK maps obtained by thresholding. Bottom: refined H, K and HK maps.

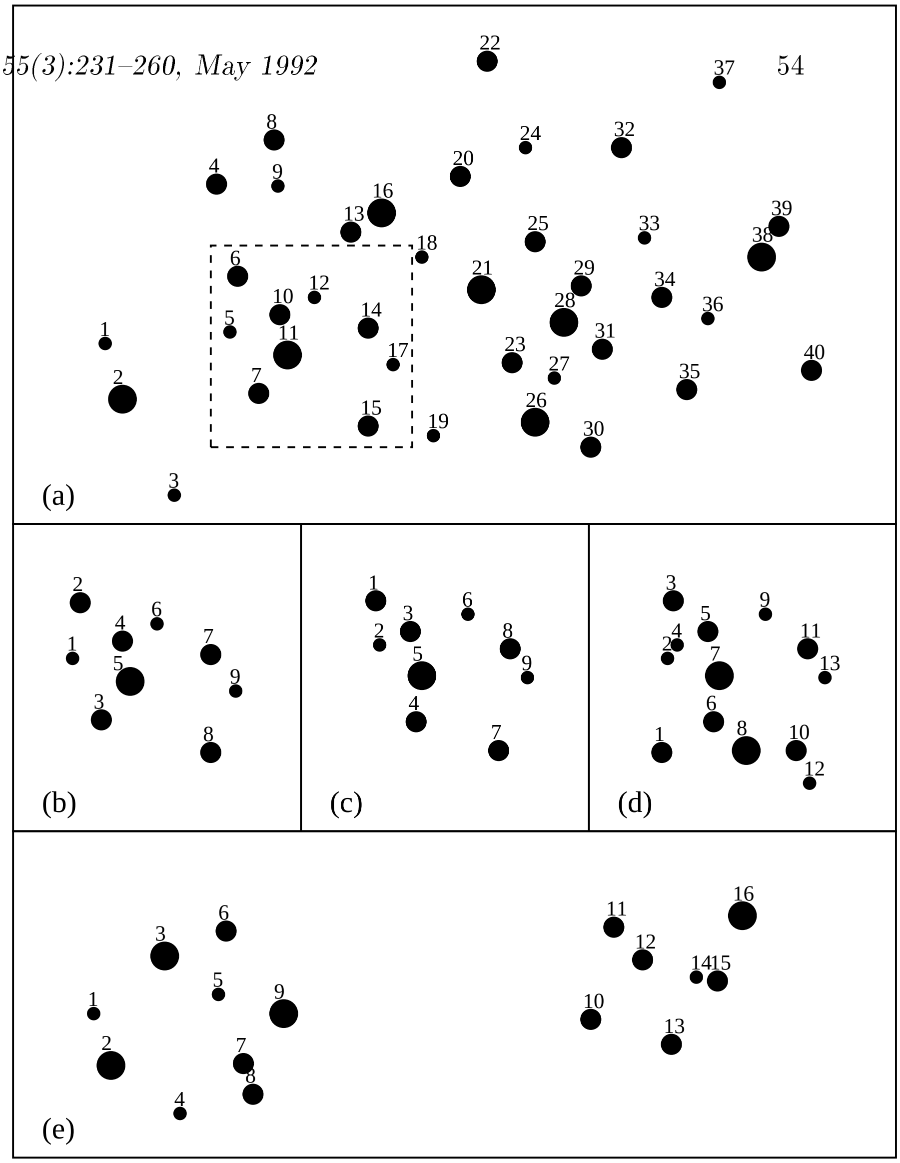


Figure 10: ARGs for inexact matching. (a) Model ARG. (b) A sub-ARG from (a). (c) A deviated version of (b). (d) Scene ARG created by adding nodes to (c). (e) Another Scene ARG (see text for its creation).

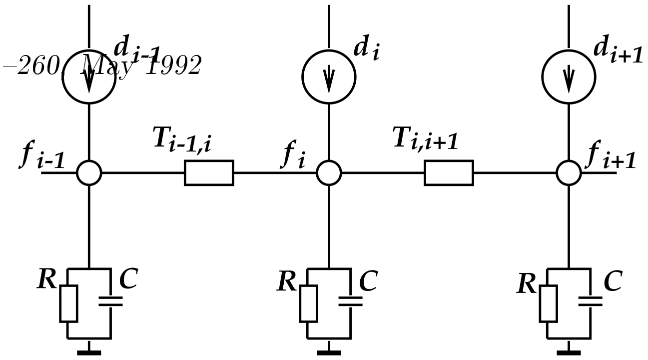


Figure 11: Analog network for low level minimization.

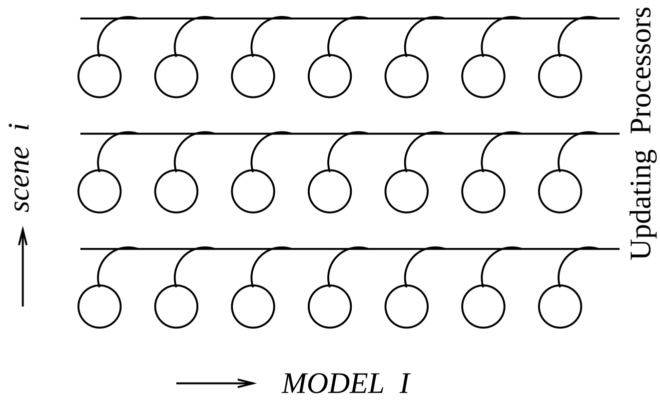


Figure 12: Schematic illustration of the labeling networks for the intermediate and high level computation.

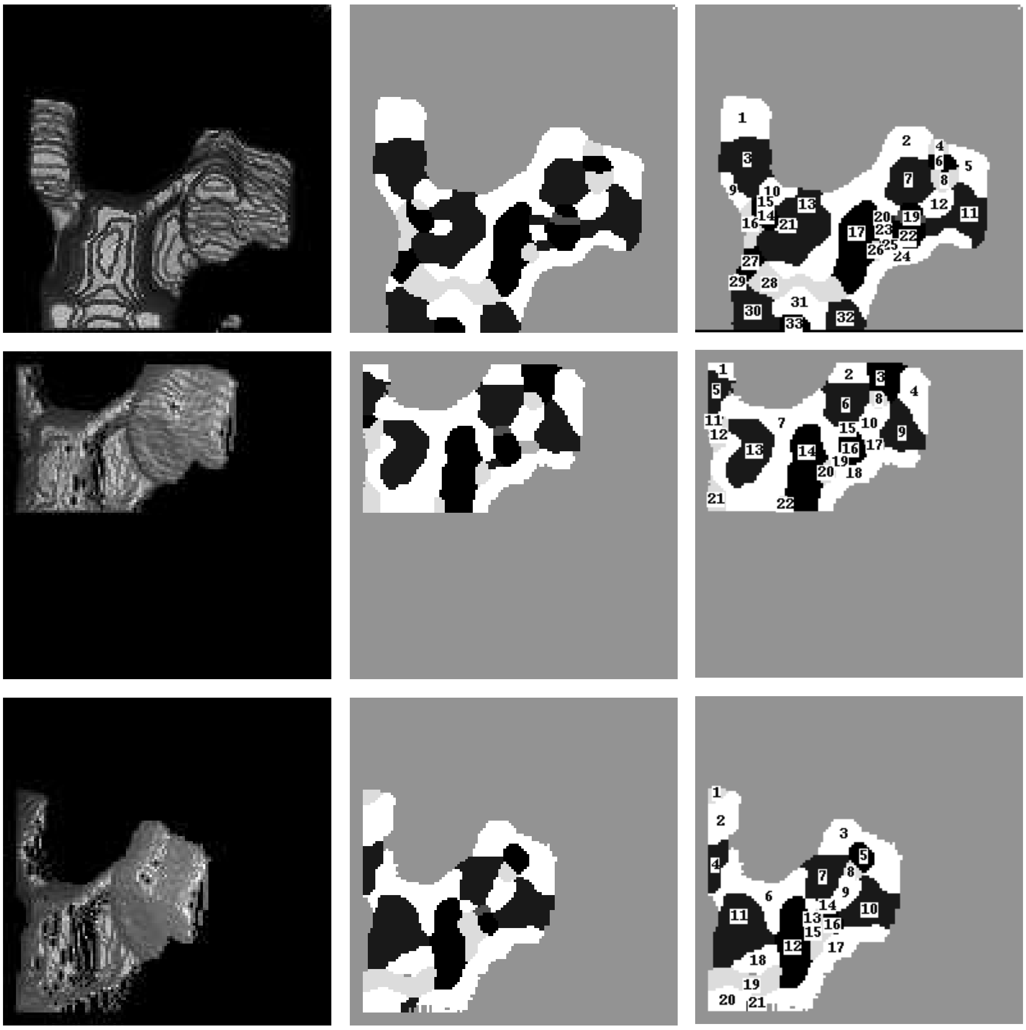


Figure 13: Model and scenes for inexact surface matching. Top: "Ground truth" model of the Renault part. Middle: Scene-1 created by a 30 degree rotation. Bottom: Scene-2 created by a 45 degree rotation.