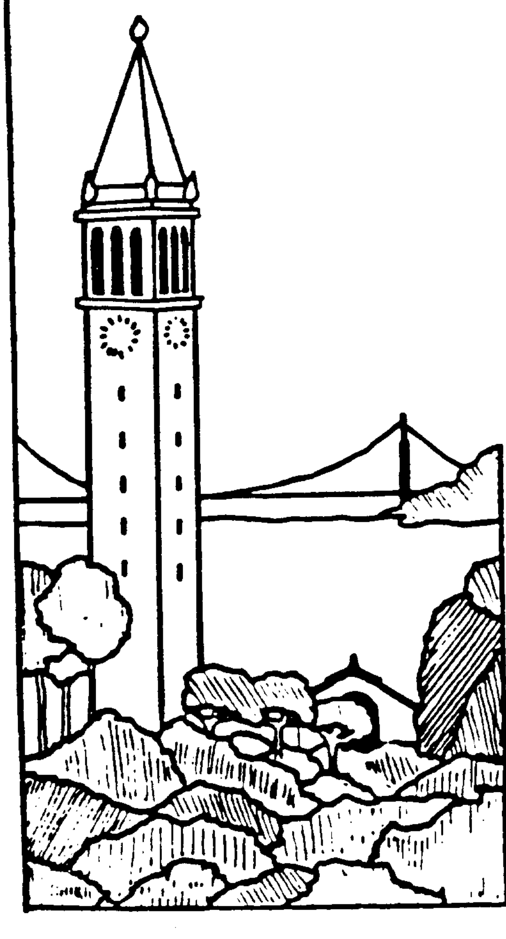
Recovering Three Dimensional Shape from a Single Image of Curved Objects

Jitendra Malik



Report No. UCB/CSD 87/340

January 1987
Computer Science Division (EECS)
University of California
Berkeley, California 94720

Recovering three dimensional shape from a single image of curved objects

Jitendra Malik

Computer Science Division, EECS
University of California, Berkeley, CA 94720

Keywords: 3-D shape recovery, shape from shading, line drawing interpretation, variational approach.

ABSTRACT

We present an algorithm to recover three-dimensional shape i.e. surface orientation and relative depth from a single segmented image. It is assumed that the scene is composed of opaque regular solid objects bounded by piecewise smooth surfaces with no markings or texture. Also it is assumed that the reflectance map $\mathbf{E}=\mathbf{R}(\mathbf{n})$ is known. For the canonical case of lambertian surfaces illuminated by a point light source, this implies knowing the light source direction.

Solutions for simplified versions of this problem have been proposed by Sugihara for the case of polyhedra, and by Horn et al for the case of a single smooth surface patch given the surface orientation on the patch boundary. This work presents the first solution for the general case.

A variational formulation of line drawing and shading constraints in a common framework is developed. Finding the 3–D shape which best satisfies these constraints is a difficult global optimization problem. The problem is made tractable by precomputing line labels at junctions using the algorithm developed in [Malik '85]. The global constraints are partitioned into constraint sets corresponding to the faces, edges and vertices in the scene. For a face the constraints are given by Horn's image irradiance equation. We develop a variational formulation of the constraints at an edge–both from the known direction of the image curve corresponding to the edge and the shading. The associated Euler-Lagrange differential equation completely captures the local information. At a vertex, the constraints are modelled by a system of non-linear equations.

An algorithm has been developed to solve this system of constraints. We are now studying various refinements intended to make the algorithm robust in the presence of image noise.

1 Introduction

In this paper, we study the problem of recovering the three–dimensional shape of the visible surfaces in a scene from a single two–dimensional image. We restrict our attention to scenes composed of opaque solid objects bounded by piecewise smooth surfaces with no markings or texture on them. By three-dimensional shape we mean a map of surface normal vectors—or equivalently relative depth—along the lines of Marr's 2 $\frac{1}{2}$D sketch, Horn's needle diagram, or Barrow and Tennenbaum's intrinsic images proposal.

As in the dominant paradigm of computational vision, the first stage of processing is edge detection resulting in the construction of a line drawing. The decomposition induced by the line drawing is referred to as the segmented image. Currently available edge detectors operating on a real image would typically result in missing lines, spurious lines, missing and improperly classified junctions. However we will ignore the resulting difficulties and assume an idealized result.

The two sources of information about 3–D shape in a segmented image are (a) the line drawing, and (b) the pixel brightness values. Both have been the subject of considerable research in computational vision. Work on line drawing interpretation of scenes containing curved objects has been most successful in qualitative characterization, e.g., in terms of line labels [16] or sign of gaussian curvature [13]. Attempts to obtain numerical surface orientation information by making additional assumptions, e.g. Brady and Yuille's [2] search for the shape maximizing the ratio of area to perimeter squared, have been successful only on a few well-chosen examples. Other work on determining numerical shape includes that of Barrow and Tennenbaum [1], and work on qualitative characterization includes that of Stevens [21].

Use of pixel brightness values in a single smooth surface patch has been the theme of the shape–from–shading work of Horn and his colleagues [9], [11], [8], [3]. Here the goal has been to solve the image irradiance equation–an equation relating surface orientation to brightness–by supplying the surface normal direction along the boundary of the patch. It is possible to do this if the patch happens to be bounded by limbs (called occluding contours by some authors) which makes the boundary surface normal calculation easy. No use is made of other kinds of lines e.g. projections of edges(tangent plane discontinuities). Consequently the approach cannot be directly employed on images of general piecewise smooth curved objects.

The question which can then be posed is—are there any schemes which try to exploit all the information in a segmented image, i.e., both line drawing and shading constraints, in order to recover quantitative 3–D shape. This has been done for the special case of polyhedra by Sugihara [23]. He represented the line drawing information as a system of linear constraints and incorporated the shading (one value for each face) information in an objective function to be minimized subject to the linear constraints. For scenes containing curved objects, the problem is significantly harder and Sugihara's approach does not have a natural generalization. To the best of our knowledge there is yet no method for 3–D shape recovery for the general class of curved objects bounded by piecewise smooth surfaces.

In this paper, we develop such a method. The algorithm builds upon, and exploits, past work on line drawing interpretation—specifically the line labelling work of Malik [16], [15], [17],—and the work of Horn and his colleagues on shape–from–shading. Shape–from–shading is reviewed in Section 2. In section 3 we analyze the constraints from a line drawing on solid shape, and point out the importance of line labels for this analysis. In sections 4 and 5 a representation of line drawing and shading constraints in a common framework is developed. In sections 6–10, we develop an algorithm for recovering the three dimensional shape which best satisfies these constraints. Some examples of objects that our scheme can handle may be found in Figures 6, 7 and 8 in Section 10.

2 Shape from Shading

The shape-from-shading problem is the problem of recovering numerical shape of a single smooth surface patch given

1. The brightness value $E(x, y)$ at each pixel
2. The reflectance map $R(\mathbf{n})$ which specifies the radiance of a surface patch as a function of its orientation.
3. The direction of the surface normal along the boundary of the patch

Note that the reflectance map encodes information about the reflecting properties of the surface and the distribution and intensity of the light sources. In the case of a lambertian surface illuminated by a point light source in the direction $\mathbf{s}$, the reflectance map $R(\mathbf{n}) = \mathbf{n} \cdot \mathbf{s}$.

The first study of shape-from-shading was by Horn [9]. There a partial differential equation relating surface elevation to image brightness was solved by converting it to an equivalent set of characteristic strip ODEs. Numerical solution of the discrete approximations of these equations had difficulty with unavoidable noise in image data. In order to address this and other difficulties, Ikeuchi and Horn [11] developed an alternative formulation of shape-from-shading as that of finding a surface orientation field which minimizes a certain functional. While Ikeuchi and Horn used sterographic coordinates f and g to specify surface orientation, subsequently Horn and Brooks [8], [3] reworked this formulation using the unit surface normal vector $\mathbf{n}$ directly.

The goal is to compute the surface normals $\mathbf{n}(x, y)$ from the (possibly noisy) brightness data $E(x, y)$. Enforcing the image irradiance equation $R(\mathbf{n}(x, y)) = E(x, y)$ gives one constraint. The second is obtained by assuming the surface to be smooth: points which are physically close to each other will have similar normal vectors. There are several ways of defining *smoothness*. Brooks and Horn [3] chose $\mathbf{n}_x^2 + \mathbf{n}_y^2$, the sum of the squares of the directional derivatives of $\mathbf{n}$ in the x and y directions. This may be viewed as a regularization[1] term [20], [24] intended to select one among a possibly infinite set of solutions.

[1] The term regularization is used here in a somewhat loose fashion following Poggio et. al.[20]. It does not seem to be the precise definition as introduced by Tikhonov. [24]

This gives the composite functional

$$\iint \{E(x,y) - R(\mathbf{n}(x,y))\}^2 + \lambda(\mathbf{n}_x^2 + \mathbf{n}_y^2) + \mu(x,y)(\mathbf{n}^2 - 1)dx\,dy$$

Determining the Euler equation and then discretizing enabled Brooks and Horn to obtain the scheme

$$\begin{cases} \mathbf{m}_{ij}^{k+1} & = & \overline{\mathbf{n}}_{ij}^{k} + \frac{\epsilon^2}{4\lambda}(E_{ij} - R(\mathbf{n}_{ij}))R_{\mathbf{n}}(\mathbf{n}_{ij}) \\ \mathbf{n}_{ij}^{k+1} & = & \mathbf{m}_{ij}^{k+1} / \parallel \mathbf{m}_{ij}^{k+1} \parallel \end{cases}$$

3 Constraints from a line drawing

In order to study the constraints imposed by a line drawing on the shape of the scene, it is convenient first to obtain a qualitative characterization of each line i.e. its 'label'. Image curves which have different line labels correspond to different constraints on 3–D shape.

3.1 Line labelling–definition and notation

Each point on an image curve in a drawing can have one of 6 possible *labels* which provide a qualitative characterization of three–dimensional physical shape at the point in the scene.

1. A "+" label represents a convex edge–an orientation discontinuity such that the two surfaces meeting along the edge in the scene enclose a filled volume corresponding to a dihedral angle greater than π.

2. A "−" label represents a concave edge–an orientation discontinuity such that the two surfaces meeting along the edge in the scene enclose a filled volume corresponding to a dihedral angle less than π.

3. A "←" or a "→" represents an occluding convex edge. When viewed from the camera, both the surface patches which meet along the edge lie on the same side, one occluding the other. As one moves in the direction of the arrow, these surfaces are to the right.

4. A "←←" or a "→→" represents a limb. Here the surface curves smoothly around to occlude itself. As one moves in the direction of

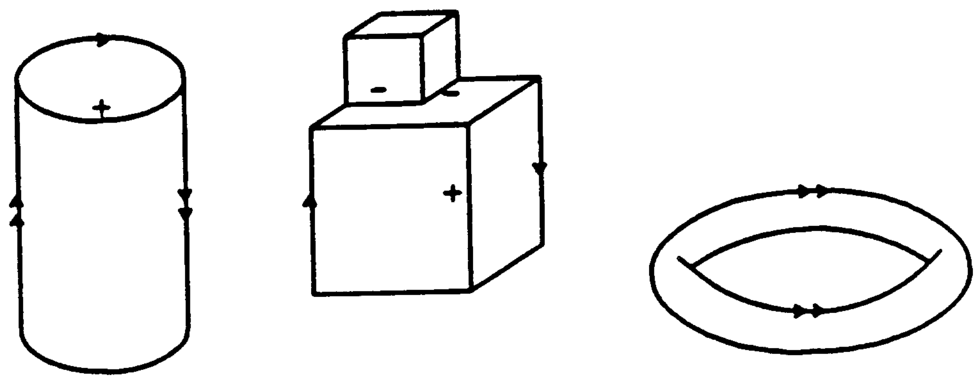


Figure 1: Different kinds of Line Labels

the twin arrows, the surface lies to the right. The line of sight is tangential to the surface for all points on the limb. Limbs move on the surface of the object as the viewpoint changes.

We will use the term *connect edge* to mean either a convex or concave edge such that both the surfaces meeting along the edge are visible. The notation for different kinds of labels is illustrated in Figure 1.

In line drawings of polyhedral scenes, the label is necessarily the same at all points on a single line segment. This permits us to use the term 'line label' as opposed to label at a point on a line. For curved objects the label can change along a line as can be seen on edge AB in Figure 2. Because of this phenomenon, we need to distinguish between two different senses of line labelling.

A *dense labelling* is a function which maps the set of *all* points on curves in the drawing into the set of labels. The dense labelling problem is to find all the dense labellings of a drawing which can correspond to a projection of some scene. Such a dense labelling is said to be *legal*. The set of legal dense labellings can be infinite (even uncountably infinite).

Instead of trying to find the label at each point on a curve, we could restrict our attention to sufficiently small neighborhoods of the junctions of the line drawing. For each line segment (between junctions) we now have to specify only two labels–one at each end. Of the 6^{2n} combinatorially possible label assignments to the n lines in a drawing only a small subset correspond to physically possible scenes. We refer to these as *legal sparse labellings.* The determination of all legal sparse labellings of a particular line drawing is the sparse labelling problem. Note that the set of legal sparse labellings is always a finite set (usually small). Figure 2 shows the legal sparse labellings of a curved object as computed by the algorithm in Malik [16]. Note that these labellings correspond to intuitive interpretations–the object floating in air, stuck to a wall or resting on a table.

In the case of polyhedra, a legal sparse labelling uniquely determines a legal dense labelling and vice versa. For drawings of curved objects, the set of dense labellings can be partitioned into equivalence classes where each equivalence class corresponds to a single sparse labelling.

Currently no algorithm is known for finding the dense labelling of a line drawing of curved objects using only the information available in a line drawing. In the context of Figure 2, it means that there is no algorithm for exactly locating X, the point of transition from convex to occluding. The sparse labelling problem has been tackled successfully–by Huffman[10] and Clowes [5] for trihedral objects, by Mackworth [14] and Sugihara [22] for arbitrary polyhedra, and by Malik [16],[15], [17] for curved objects.

3.2 How does a line labelling constrain solid shape?

Lines with different labels correspond to different types of constraints. In this section we study the system of position and orientation constraints associated with a dense labelling of a line drawing. Some of these constraints are well known and have been used before; some others have been expressed in gradient-space (p, q) notation. Our purpose is to provide a coherent list of the 'fundamental' constraints using unit surface normal vectors. The motivation is to be able to consider line drawing and shading constraints in a common framework. Gradient-space coordinates are inadequate for representing surface orientation for curved objects with limbs.

It is assumed that the line drawing has been formed by orthographic

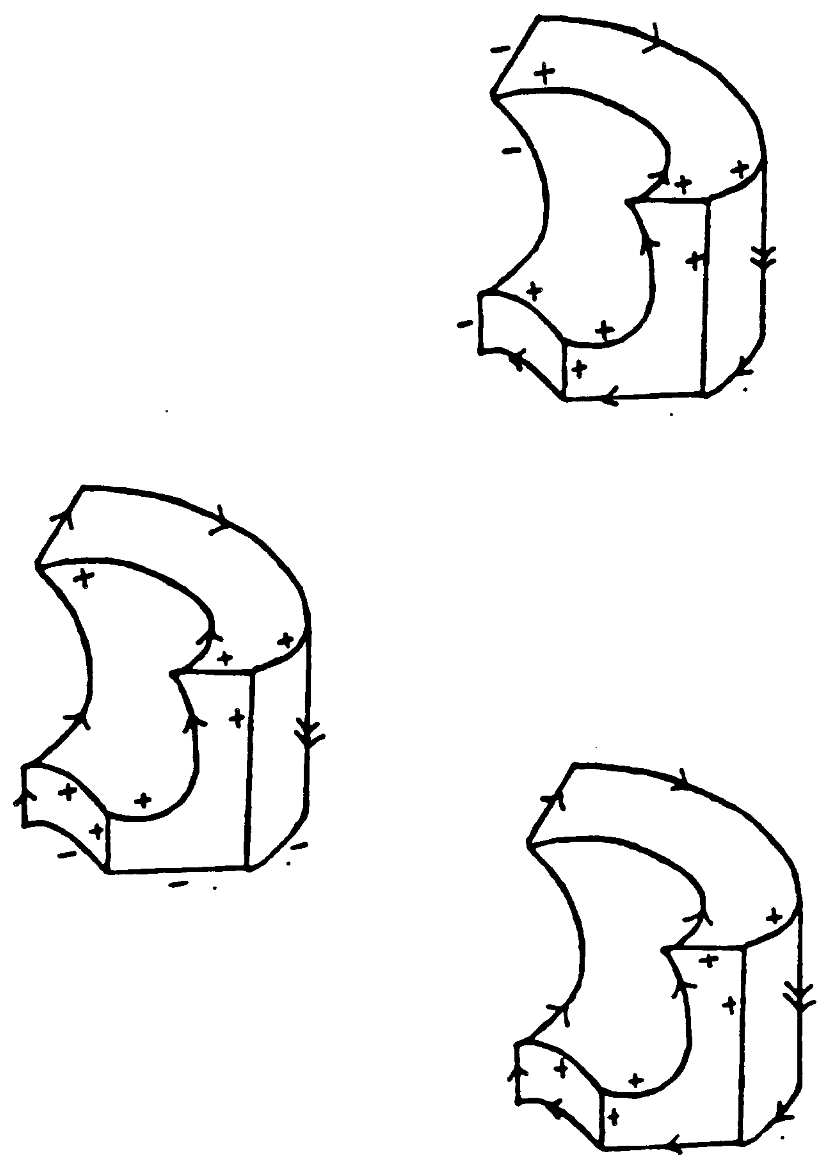


Figure 2: Legal labellings of a curved object

projection, with the eye along the z-axis at $z = +\infty$. We now consider the constraints from the different elements of a labelled line drawing.

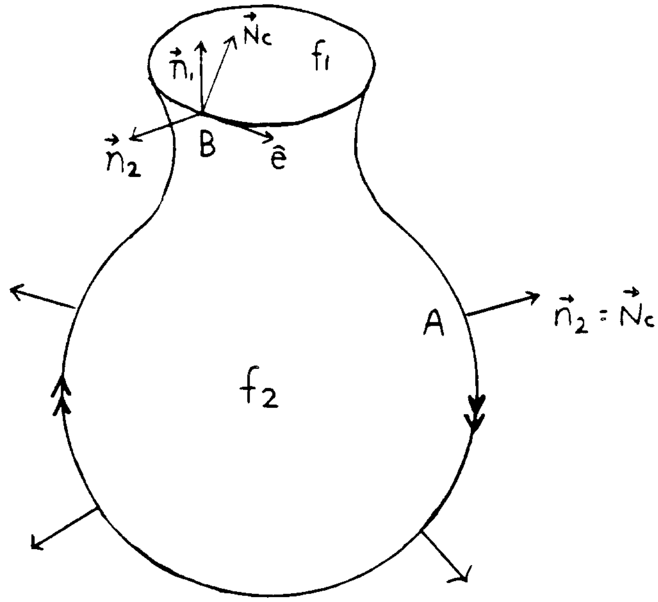
1. *Shape constraint at a limb:* At limbs one can determine the surface orientation uniquely. Let $\mathbf{n}$ be the unit surface normal, and $\mathbf{l}$ the unit tangent vector at a point on the limb. Obviously, $\mathbf{n}.\mathbf{l} = \mathbf{0}$. As a limb corresponds to points on the surface where the line of sight vector $\hat{\mathbf{z}}$ lies in the tangent plane to the surface, we also have $\mathbf{n}.\hat{\mathbf{z}} = \mathbf{0}$ for points on the limb(equivalently $n_z = 0$). $\mathbf{n}$ therefore lies in the image plane and is in the direction of $\mathbf{N}_c$ — the outward pointing unit vector in the image plane drawn perpendicular to the projection of the limb. See Figure 3.

   What is stated above is the orientation constraint for the surface on which the limb lies i.e. the surface on the right of the twin arrows. We also have a position constraint—the surface on the right of the twin arrows is nearer, implying a linear inequality between the z values on either side of the limb.

   The surface orientation constraint due to limbs is well-known and has been used in [1], [11], [8].

2. *Shape constraint at an edge:* Let $\hat{\mathbf{e}}$ be the unit tangent vector to the edge at a point, and let $\mathbf{n}_1$ and $\mathbf{n}_2$ be the unit surface normals to the tangent planes to the two faces f_1 and f_2 at the point. Let $\hat{\mathbf{e}}$ be oriented such that when one walks on the edge in the direction of $\hat{\mathbf{e}}$, the face f_1 is to the left (see Figure 3). Now $\hat{\mathbf{e}}$ is perpendicular to $\mathbf{n}_1$ because $\hat{\mathbf{e}}$ lies in the tangent plane to the face f_1. Similarly $\hat{\mathbf{e}}$ is perpendicular to $\mathbf{n}_2$. Therefore $\hat{\mathbf{e}}$ is parallel to $\mathbf{n}_1 \times \mathbf{n}_2$. We do not know the vector $\hat{\mathbf{e}}$, but from a line drawing we can determine its orthographic projection into the image plane. We thus have the constraint $(\mathbf{n}_1 \times \mathbf{n}_2)_{\mathrm{proj}} = \lambda \hat{\mathbf{e}}_{\mathrm{proj}}$. Here the notation $\mathbf{v}_{\mathrm{proj}}$ is used for the orthographic projection of $\mathbf{v}$ into the image plane. λ is a positive scalar if the edge is convex, negative if the edge is concave. Note that this constraint is equally valid for occluding convex edges, where one of the surface normals corresponds to a hidden face.

   This constraint when expressed using p, q–the gradient space coordinates–gives the rule used by Mackworth [14] and many other researchers in their gradient space constructions.



AT A $\quad \vec{n}_2 = \vec{N}_c$

AT B $\quad \vec{n}_1 \times \vec{n}_2 = \lambda \hat{e}, \quad [n_1 \; n_2 \; N_c] = 0$

Figure 3: Orientation constraints at limbs and edges

The position constraint at an edge is trivial—the depth z is continuous across a convex or concave edge and is discontinuous at an occluding edge.

For later use, it is convenient to develop an alternative version of the orientation constraint. Let $\mathbf{N}_c$ be a unit vector in the image plane perpendicular to $\hat{\mathbf{e}}_{\text{proj}}$. As $(\mathbf{n}_1 \times \mathbf{n}_2)_{\text{proj}} = \lambda \hat{\mathbf{e}}_{\text{proj}}$, it follows that

$$(\mathbf{n}_1 \times \mathbf{n}_2)_{\text{proj}} \cdot \mathbf{N}_c = 0$$

As $\mathbf{N}_c$ has no component in the z-direction, this is equivalent to saying that $(\mathbf{n}_1 \times \mathbf{n}_2) \cdot \mathbf{N}_c = 0$, or using the vector triple product notation

$$[\mathbf{n}_1\, \mathbf{n}_2\, \mathbf{N}_c] = 0$$

3. *Shape constraint inside an area:* If each area in the image is to be the projection of a connected part of a smooth surface, the functions that map each image point to its position and orientation must be smooth within a single area. Also the surface normals at all the visible surface patches must have positive n_z components.

   Note that as the surface normal in a smooth patch can be written in terms of the partial derivatives of z with respect to x and y, these two functions are not independent. If we specify a C^2 function $z(x, y)$, the orientation function $\mathbf{n}(x, y)$ is automatically determined and smooth.

We feel that it is appropriate to regard the position and orientation constraints listed above as the fundamental system of constraints associated with a dense labelling of a line drawing. We'll refer to this constraint set as the DL–system corresponding to a dense labelling. A candidate solution to this set of constraints is obtained by specifying

1. A piecewise smooth function $z(x, y)$ corresponding to the depth at each visible point in the scene.

2. A smooth function $\mathbf{n}_h(x, y)$ defined on all points on the lines in the drawing which are labelled $\leftarrow$ or $\rightarrow$. This function corresponds to the surface normal on the hidden face at that point on the occluding edge.

As pointed out earlier, the surface normal at each visible point is then automatically determined.

It is obvious that for a dense labelling of a line drawing to correspond to a legal scene, it is necessary that there exist a candidate solution which satisfies its DL–system.

4 Recovering 3–D shape given a dense labelling

If the dense labelling of a line drawing is known, the problem of three–dimensional shape recovery becomes that of finding a shape such that:

1. It satisfies the constraints in the DL–system.

2. In the interior of each surface patch, the image irradiance equation $E(x,y) = R(\mathbf{n}(x,y))$ is satisfied.

We adopt the following strategy:

1. Find the surface normal $\mathbf{n}(x,y)$ using the orientation constraints.

2. Break up the image into components which correspond to connected surfaces in the scene. In each component, depth can be found by integration. Note that the solution for each component will have an as yet undetermined constant of integration.

3. The position constraints in the DL–system can be used to find a set of linear inequalities among these constants of integration. If a feasible solution exists, we are done.

The problem of finding the surface normal can be formulated as that of finding a piecewise smooth function $\mathbf{n}(x,y)$ which minimizes the sum of the following four terms:

- $\iint_I (E(x,y) - R(\mathbf{n}(x,y)))^2 \, dx \, dy$ which measures the error in satisfying the image irradiance equation. Here I is the entire image.

- $\int_E [\mathbf{n}_1(s)\,\mathbf{n}_2(s)\,\mathbf{N}_c(s)]^2\,ds$ where E is the set of image curves which have been labelled as convex or concave edges and s is the arc length parameter along the edge. $\mathbf{n}_1(s)$ and $\mathbf{n}_2(s)$ are the two limits of the discontinuous function $\mathbf{n}(x,y)$ when a point on an edge is approached on its two faces. This term measures the error in satisfying the edge constraint.

- $\int_L (\mathbf{n}(s) - \mathbf{N}_c(s))^2$ ds where E is the set of all limbs. $\mathbf{N}_c(s)$ has a sense determined by the direction of the line labelling–if one walks on the limb in the direction of the double arrows, $\mathbf{N}_c(s)$ points to the left. See Figure 3.

- $\iint_{I-B} (\mathbf{n}_x^2(x,y) + \mathbf{n}_y^2(x,y))\,dx\,dy$ which is a 'regularization term' intended to select a particularly smooth solution. Here B denotes the set of all curves in the line drawing.

The weighted sum of these terms gives us the composite functional

$$\lambda_1 \iint_I (E - R(\mathbf{n}))^2\,dx\,dy + \lambda_2 \int_E [\mathbf{n}_1\,\mathbf{n}_2\,\mathbf{N}_c]^2\,ds + \lambda_2 \int_L (\mathbf{n} - \mathbf{N}_c)^2\,ds + \lambda_3 \iint_{I-B} (\mathbf{n}_x^2 + \mathbf{n}_y^2)\,dx\,dy$$

One could find the Euler equation and develop a numerical scheme for minimizing this functional–however that would be a futile exercise as there is no available algorithm for determining the dense labelling in advance. Indeed one might argue that the determination of the dense labelling occurs as part of the process of trying to satisfy the image constraints. This approach is developed further in the next section.

5 Simultaneous recovery of 3–D shape and the dense labelling

We now drop the unrealistic assumption that the dense labelling is known in advance, but instead seek to recover it as part of the process of finding the shape $\mathbf{n}(x,y)$. Introduce binary (0 or 1) valued 'indicator functions' e_1, e_2, l_1, and l_2 that code the label at a point on the line drawing as follows:

Let f_1 and f_2 be the two faces corresponding to the two areas bordering the image curve c. Then

$$\begin{aligned}
e_1 &= 1 && \text{iff } c \text{ is an edge on } f_1 \\
e_2 &= 1 && \text{iff } c \text{ is an edge on } f_2 \\
l_1 &= 1 && \text{iff } c \text{ is a limb on } f_1 \\
l_2 &= 1 && \text{iff } c \text{ is a limb on } f_2
\end{aligned}$$

It may be observed that $e_1 e_2 = 1$ iff c is a convex or concave edge. Also if c is a limb, then exactly one of l_1, l_2 is equal to 1–the direction of the double arrows determines which one.

Using these indicator functions we can formulate shape recovery as a problem of finding $\mathbf{n}(x, y)$, $e_1(s)$, $e_2(s)$, $l_1(s)$, $l_2(s)$ such that the following functional is minimized:

$$\lambda_1 \iint_I (E - R(\mathbf{n}))^2 \, dx \, dy + \lambda_2 \int_B e_1 e_2 [\mathbf{n}_1 \, \mathbf{n}_2 \, \mathbf{N}_c]^2 + l_1 (\mathbf{n}_1 - \mathbf{N}_c)^2 + l_2 (\mathbf{n}_2 + \mathbf{N}_c)^2 \, ds$$
$$+ \lambda_3 \iint_{I-B} (\mathbf{n}_x^2 + \mathbf{n}_y^2) \, dx \, dy \quad (1)$$

subject to the constraints

$$\begin{aligned}
\forall s. e_1(s) + e_2(s) + l_1(s) + l_2(s) &\geq 1 \\
\forall s. e_1(s) + l_1(s) + l_2(s) &\leq 1 \\
\forall s. e_2(s) + l_1(s) + l_2(s) &\leq 1
\end{aligned}$$

and, of course the constraint that $\mathbf{n}^2 = 1$.

The side constraints enforce the basic properties of line labels–a curve can not be both a limb and an edge etc. One can generalize this idea to incorporate constraints due to restrictions on local labelling possibilities at junctions.

To the best of our knowledge, the above formulation is the first to attempt to include all the constraints in a single image of curved objects in a coherent framework. For polyhedral objects Sugihara [23] has solved this problem, but for curved objects shading and line drawing constraints had not previously been studied in a common framework.

Trying to minimize globally the functional (1) is a formidable task. Here are two possible approaches:

1. *Gradient Descent:* Starting from some initial guessed shape, iteratively adjust $\mathbf{n}$, e_1, e_2, l_1, l_2 so as to lower the cost measured by (1). One could use algorithms from optimization theory or code this problem onto a Hopfield 'neural' network [7] by suitably defining T_{ij}. The hard part of course is in coming up with an initial guess good enough to avoid local minima. A modification of this approach which has been tried by Witkin et al [27] is to introduce a scale parameter σ and thus embed the problem in a larger space. Gradient descent is first used to solve a smoothed version of the minimization problem and then track the minimum continuously as σ tends to zero. While there is no guarantee that the global minimum will be found, Witkin et al found their empirical results satisfactory.

2. *Simulated Annealing:* We refer here to the class of global optimization algorithms inspired by ideas from statistical physics e.g. Kirkpatrick et al [12] and Geman and Geman [6]. By allowing steps which may increase energy, the system can avoid getting trapped in local minima–the bane of simple-minded gradient descent. However the computational costs of these algorithms are extremely high.

Clearly further work is needed to flesh out these approaches. In this paper we will develop a third approach based on precomputing the sparse labelling.

6 Recovering 3–D shape using a partitioned constraint set approach

The problem of recovering three–dimensional shape from a single image becomes more tractable if one notes the following:

- The set of legal *sparse* labellings of the drawing can be determined *before* examining in detail the constraints associated with possible DL–systems and the shading. An algorithm for doing this, based on constructing a catalog of legal labelling possibilities for each kind of junction, was developed in [16]. The scheme developed in the rest of this paper exploits this.

- The objective of the global minimization suggested in Sections 4 and 5 is to attempt to satisfy the shading and line drawing constraints over the *entire* image simultaneously. Instead of trying to do it in one fell swoop, one could partition these constraints into the following three classes:

  1. Constraints in the interior of an image area.
  2. Constraints in the neighborhood of an image curve.
  3. Constraints in the neighborhood of a junction.

  Surface continuity requirements provide the cross–coupling among the solutions of these constraint sets.

In the next three sections we do the following analysis for each type of constraint set:

- Formulate the constraints.
- Examine the conditions for unique solvability.
- Develop a numerical scheme for finding the solution.

Finally in section 10, we study how local shape recovery procedures can be combined to find the global shape.

7 Shape constraints inside an area

The only source of constraint is the shading and we wish to find the 'smoothest' surface consistent with the pixel brightness values. This is exactly the shape from shading problem which has been studied by Horn and his colleagues and we can adopt their analyses and algorithms unchanged. Specifically, we use the relaxation scheme suggested by Horn and Brooks which is stated in section 2 of this paper.

The question of the existence of a unique solution for (a) the shape-from-shading problem, and, (b) the numerical scheme, does not seem to have a clean theoretical answer–though partial results were obtained by Bruss in [4]. Simulation results e.g. [11] indicate no problems in practice.

The Horn–Brooks relaxation scheme, just like the other algorithms for this problem, requires knowledge of the direction of the surface normal along the boundary of the patch. Empirical evidence suggests that knowing the surface normal along a significant fraction of the boundary is adequate in practice.

8 Shape constraints along an image curve

At a limb, the surface normal is uniquely determined by the direction of the projected curve. So the interesting case is that of an edge, where there are both shading and line drawing constraints. We want to use these to determine the surface normals $\mathbf{n}_1(s)$ and $\mathbf{n}_2(s)$ on the two faces f_1 and f_2 as a function of the arc length parameter s along the edge. To do this, we have to measure the brightness values $E_1(s)$, $E_2(s)$ on the two sides of the edge in the line drawing. As edges are typically not neatly aligned with pixel boundaries, this would entail taking the weighted average of neighbouring pixels which lie entirely on a single face. As defined in section 3, $\mathbf{N}_c(s)$ is a unit length vector drawn normal to the projection of the edge in the image plane.

First we do some equation counting. To determine $\mathbf{n}_1$ and $\mathbf{n}_2$ one has to solve for 4 independent parameters. (While there are 3 components of $\mathbf{n}$, the unit length constraint means that only two are independent.) If the edge is connect (convex or concave), there are two equations due to shading: $E_1 = R(\mathbf{n}_1)$ and $E_2 = R(\mathbf{n}_2)$. The direction of the edge gives us one more equation $[\mathbf{n}_1\,\mathbf{n}_2\,\mathbf{N}_c] = 0$, still leaving us one equation short. We conclude that it is not possible to solve for $\mathbf{n}_1$, $\mathbf{n}_2$ by just looking at the neighborhood of a point on an edge, just as in the standard shape-from-shading problem it is not possible to compute $\mathbf{n}$ locally. We need to make use of boundary conditions, and maximize smoothness–points which are physically close should have similar surface normals .

To satisfy the image irradiance equation on both faces, we seek to minimize

$$\int (E_1(s) - R(\mathbf{n}_1(s)))^2 + (E_2(s) - R(\mathbf{n}_2(s)))^2\, ds$$

To satisfy the constraint due to the observed direction of the projection of the edge, minimize

$$\int [\mathbf{n}_1(s)\,\mathbf{n}_2(s)\,\mathbf{N}_c(s)]^2\,ds$$

Next we add a 'regularization term' by incorporating the expression

$$\int (\mathbf{n}_1'(s))^2 + (\mathbf{n}_2'(s))^2\,ds$$

Here $\mathbf{n}_1'(s)$, $\mathbf{n}_2'(s)$ are the derivatives of $\mathbf{n}_1(s)$ and $\mathbf{n}_2(s)$ with respect to arc length. Finally we insist that the normals $\mathbf{n}_1(s)$ and $\mathbf{n}_2(s)$ have unit length. Combining these constraints we have the composite functional

$$\begin{aligned}\int \lambda_1\{(E_1 - R(\mathbf{n}_1))^2 + (E_2 - R(\mathbf{n}_2))^2\} + \lambda_2[\mathbf{n}_1\,\mathbf{n}_2\,\mathbf{N}_c]^2 \\ +\lambda_3\{(\mathbf{n}_1')^2 + (\mathbf{n}_2')^2\} + \mu_1(\mathbf{n}_1^2 - 1) + \mu_2(\mathbf{n}_2^2 - 1)\,ds \qquad (2)\end{aligned}$$

$\mu_1(s)$, $\mu_2(s)$ are Lagrange multipliers.
This is a functional of the form

$$\int F(s, \mathbf{n}_1, \mathbf{n}_2, \mathbf{n}_1', \mathbf{n}_2')ds$$

which gives rise to the Euler equations

$$F_{\mathbf{n}_1} - \frac{d}{ds}F_{\mathbf{n}_1'} = 0$$
$$F_{\mathbf{n}_2} - \frac{d}{ds}F_{\mathbf{n}_2'} = 0$$

and the 'natural' boundary conditions $F_{\mathbf{n}_1'} = 0$ and $F_{\mathbf{n}_2'} = 0$.

It follows that the edge constraint functional (2) has the Euler equations

$$-2\lambda_1(E - R(\mathbf{n}_1))R_{\mathbf{n}_1} + 2\lambda_2[\mathbf{n}_1\,\mathbf{n}_2\,\mathbf{N}_c](\mathbf{n}_2 \times \mathbf{N}_c) - 2\lambda_3\mathbf{n}_1'' + 2\mu_1\mathbf{n}_1 = 0 \quad (3)$$
$$-2\lambda_1(E - R(\mathbf{n}_2))R_{\mathbf{n}_2} - 2\lambda_2[\mathbf{n}_1\,\mathbf{n}_2\,\mathbf{N}_c](\mathbf{n}_1 \times \mathbf{N}_c) - 2\lambda_3\mathbf{n}_2'' + 2\mu_2\mathbf{n}_2 = 0 \quad (4)$$

and the 'natural' boundary conditions $\mathbf{n}_1' = 0$ and $\mathbf{n}_2' = 0$.

We can find the Lagrange multiplier μ_1 by taking the dot product of equation (3) with $\mathbf{n}_1$ and noting that $\mathbf{n}_1 \cdot \mathbf{n}_1 = 1$

$$\mu_1 = \lambda_1(E - R(\mathbf{n}_1))R_{\mathbf{n}_1} \cdot \mathbf{n}_1 - \lambda_2[\mathbf{n}_1\,\mathbf{n}_2\,\mathbf{N}_c]^2 + \lambda_3\mathbf{n}_1'' \cdot \mathbf{n}_1$$

We can now eliminate μ_1 by substituting it back, giving

$$-\lambda_1(E - R(\mathbf{n}_1))\{R_{\mathbf{n}_1} - (R_{\mathbf{n}_1}\cdot\mathbf{n}_1)\mathbf{n}_1\}$$
$$+\lambda_2[\mathbf{n}_1\,\mathbf{n}_2\,\mathbf{N}_c]\{(\mathbf{n}_2\times\mathbf{N}_c) - [\mathbf{n}_1\,\mathbf{n}_2\,\mathbf{N}_c]\mathbf{n}_1\} - \lambda_3\{\mathbf{n}_1'' - (\mathbf{n}_1''\cdot\mathbf{n}_1)\mathbf{n}_1\} = \mathbf{0} \quad (5)$$

A similar manipulation for the other Euler equation gives

$$-\lambda_1(E - R(\mathbf{n}_2))\{R_{\mathbf{n}_2} - (R_{\mathbf{n}_2}\cdot\mathbf{n}_2)\mathbf{n}_2\}$$
$$-\lambda_2[\mathbf{n}_1\,\mathbf{n}_2\,\mathbf{N}_c]\{(\mathbf{n}_1\times\mathbf{N}_c) + [\mathbf{n}_1\,\mathbf{n}_2\,\mathbf{N}_c]\mathbf{n}_2\} - \lambda_3\{\mathbf{n}_2'' - (\mathbf{n}_2''\cdot\mathbf{n}_2)\mathbf{n}_2\} = \mathbf{0} \quad (6)$$

Note that the equation (5) is an equation in components perpendicular to $\mathbf{n}_1$ and is thus equivalent to only two independent scalar equations. This can be seen by taking the dot product of the left hand side with $\mathbf{n}_1$ and observing that it is identically $\mathbf{0}$. Similarly equation (6) is an equation in components perpendicular to $\mathbf{n}_2$.

The additional constraints that are needed come from the fact that $\mathbf{n}_1$ and $\mathbf{n}_2$ are of unit length.

Subsequent analysis will be facilitated by noting the following identities. $\mathbf{x}$ can be any vector, $\mathbf{n}$ is a unit length vector, $\mathbf{p}$ is any vector perpendicular to $\mathbf{n}$ i.e. $\mathbf{n}\cdot\mathbf{p} = 0$.

$$\begin{aligned} (\mathbf{I} - \mathbf{n}\mathbf{n}^T)\mathbf{n} &= \mathbf{0} \\ (\mathbf{I} - \mathbf{n}\mathbf{n}^T)\mathbf{p} &= \mathbf{p} \\ (\mathbf{I} - \mathbf{n}\mathbf{n}^T)\mathbf{x} &= \mathbf{x} - (\mathbf{x}\cdot\mathbf{n})\mathbf{n} \end{aligned} \quad (7)$$

Using the third of these identities and introducing $\alpha = \frac{\lambda_1}{\lambda_3}$, and $\beta = \frac{\lambda_2}{\lambda_3}$ we can transform equation (5) to

$$\begin{aligned} (\mathbf{I} - \mathbf{n}_1\mathbf{n}_1^T)\mathbf{n}_1'' &= -\alpha(E - R(\mathbf{n}_1))\{R_{\mathbf{n}_1} - (R_{\mathbf{n}_1}\cdot\mathbf{n}_1)\mathbf{n}_1\} \\ &\quad +\beta[\mathbf{n}_1\,\mathbf{n}_2\,\mathbf{N}_c]\{(\mathbf{n}_2\times\mathbf{N}_c) - [\mathbf{n}_1\,\mathbf{n}_2\,\mathbf{N}_c]\mathbf{n}_1\} \end{aligned} \quad (8)$$

In order to develop a numerical scheme, we use the discrete approximation

$$\mathbf{n}_1^{i\prime\prime} \approx \frac{\mathbf{n}_1^{i+1} - 2\mathbf{n}_1^{i} + \mathbf{n}_1^{i-1}}{(\Delta s)^2}$$

to obtain

$$(\mathbf{I} - \mathbf{n}_1^i \mathbf{n}_1^{i\,T})\left(\frac{\mathbf{n}_1^{i+1} - 2\mathbf{n}_1^i + \mathbf{n}_1^{i-1}}{(\Delta s)^2}\right) = -\alpha(E - R(\mathbf{n}_1^i))\{R^i_{\mathbf{n}_1} - (R^i_{\mathbf{n}_1} \cdot \mathbf{n}_1^i)\mathbf{n}_1^i\}$$
$$+\beta[\mathbf{n}_1^i\,\mathbf{n}_2^i\,\mathbf{N}_c^i]\{(\mathbf{n}_2^i \times \mathbf{N}_c^i) - [\mathbf{n}_1^i\,\mathbf{n}_2^i\,\mathbf{N}_c^i]\mathbf{n}_1^i\} \quad (9)$$

It is convenient to introduce notation corresponding to the components of $\mathbf{n}_1^{i-1}$ and $\mathbf{n}_1^{i+1}$ along $\mathbf{n}_1^i$ and perpendicular to $\mathbf{n}_1^i$

$$\begin{aligned} \mathbf{n}_1^{i-1} &= c_{1i}^- \mathbf{n}_1^i + \mathbf{q}_1^i \\ \mathbf{n}_1^{i+1} &= c_{1i}^+ \mathbf{n}_1^i + \mathbf{p}_1^i \end{aligned}$$

Using these substitutions for $\mathbf{n}_1^{i-1}$ and $\mathbf{n}_1^{i+1}$ in (9) and simplifying using the identities in (7), we get

$$\begin{aligned} \mathbf{p}_1^i = -\mathbf{q}_1^i &- \alpha(\Delta s)^2(E - R(\mathbf{n}_1^i))\{R^i_{\mathbf{n}_1} - (R^i_{\mathbf{n}_1} \cdot \mathbf{n}_1^i)\mathbf{n}_1^i\} \\ &+\beta(\Delta s)^2[\mathbf{n}_1^i\,\mathbf{n}_2^i\,\mathbf{N}_c^i]\{(\mathbf{n}_2^i \times \mathbf{N}_c^i) - [\mathbf{n}_1^i\,\mathbf{n}_2^i\,\mathbf{N}_c^i]\mathbf{n}_1^i\} \end{aligned} \quad (10)$$

Now $\mathbf{q}_1^i = \mathbf{n}_1^{i-1} - (\mathbf{n}_1^{i-1} \cdot \mathbf{n}_1^i)\mathbf{n}_1^i$ is known, enabling us to calculate $\mathbf{p}_1^i$. Next note that

$$\begin{aligned} \mathbf{n}_1^{i+1} \cdot \mathbf{n}_1^{i+1} &= (c_{1i}^+ \mathbf{n}_1^i + \mathbf{p}_1^i) \cdot (c_{1i}^+ \mathbf{n}_1^i + \mathbf{p}_1^i) \\ &= (c_{1i}^+)^2 + (\mathbf{p}_1^i)^2 \\ &= 1 \end{aligned}$$

which implies that

$$c_{1i}^+ = \sqrt{1 - (\mathbf{p}_1^i)^2}$$

Theoretically there are two solutions for c_{1i}^+, one positive and one negative. Obviously, the positive root is to be used — the negative root would result in a $\mathbf{n}_1^{i+1}$ in almost the opposite direction to $\mathbf{n}_1^i$.

We thus have the following scheme for computing $\mathbf{n}_1^{i+1}$ given knowledge of $\mathbf{n}_1^{i-1}$, $\mathbf{n}_1^i$ and $\mathbf{n}_2^i$:

$$\begin{aligned} \mathbf{q}_1^i &= \mathbf{n}_1^{i-1} - (\mathbf{n}_1^{i-1} \cdot \mathbf{n}_1^i)\mathbf{n}_1^i \\ \mathbf{p}_1^i &= -\mathbf{q}_1^i - \alpha(\Delta s)^2(E - R(\mathbf{n}_1^i))\{R^i_{\mathbf{n}_1} - (R^i_{\mathbf{n}_1} \cdot \mathbf{n}_1^i)\mathbf{n}_1^i\} + \\ &\quad \beta(\Delta s)^2[\mathbf{n}_1^i\,\mathbf{n}_2^i\,\mathbf{N}_c^i]\{(\mathbf{n}_2^i \times \mathbf{N}_c^i) - [\mathbf{n}_1^i\,\mathbf{n}_2^i\,\mathbf{N}_c^i]\mathbf{n}_1^i\} \qquad (11) \\ \mathbf{n}_1^{i+1} &= \mathbf{p}_1^i + \mathbf{n}_1^i\sqrt{1 - (\mathbf{p}_1^i)^2} \end{aligned}$$

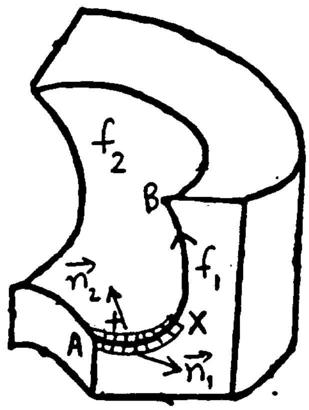


Figure 4: Dealing with edge label transitions

By an analagous analysis starting from (6) we obtain

$$
\begin{aligned}
\mathbf{q}_2^i &= \mathbf{n}_2^{i-1} - (\mathbf{n}_2^{i-1} \cdot \mathbf{n}_2^i)\mathbf{n}_2^i \\
\mathbf{p}_2^i &= -\mathbf{q}_2^i - \alpha(\Delta s)^2(E - R(\mathbf{n}_2^i))\{R_{\mathbf{n}_2}^i - (R_{\mathbf{n}_2}^i \cdot \mathbf{n}_2^i)\mathbf{n}_2^i\} - \\
&\quad \beta(\Delta s)^2[\mathbf{n}_1^i\, \mathbf{n}_2^i\, \mathbf{N}_c^i]\{(\mathbf{n}_1^i \times \mathbf{N}_c^i) + [\mathbf{n}_1^i\, \mathbf{n}_2^i\, \mathbf{N}_c^i]\mathbf{n}_2^i\} \qquad (12) \\
\mathbf{n}_2^{i+1} &= \mathbf{p}_2^i + \mathbf{n}_2^i\sqrt{1 - (\mathbf{p}_2^i)^2}
\end{aligned}
$$

If $\mathbf{n}_1^0$, $\mathbf{n}_1^1$, $\mathbf{n}_2^0$, $\mathbf{n}_2^1$ are known, these schemes tell us how to 'grow' the solution. The iteration can in fact be started just given the values of $\mathbf{n}_1$, $\mathbf{n}_2$ at the initial point. As $\mathbf{n}_1'$ and $\mathbf{n}_2'$ are both equal to 0 at the initial point (natural boundary conditions), we assume $\mathbf{n}_1^1 = \mathbf{n}_1^0$ and $\mathbf{n}_2^1 = \mathbf{n}_2^0$.

It should be pointed out that the analysis in this section has been for connect edges—if the edge is occluding, we do not know the brightness values for the occluded face and thus have one less constraint. For curved objects, an edge can change its label from convex to occluding between junctions e.g. at point X in Figure 4. However we know [16] that such points correspond to invisible limbs, and therefore can be detected by checking if $\mathbf{n}_z$ is sufficiently small. In Figure 4, $\mathbf{n}_{2z} = 0$ at X. The iterations in (11) and (12) should be stopped at this point.

Now we consider a case when more information is available. Suppose that in addition to knowing $\mathbf{n}_1$, $\mathbf{n}_2$ at the initial point, $\mathbf{n}_2(s)$ is also known along the edge (as a result of some previous computation). In this case, just use (11).

When $\mathbf{n}_2(s)$ is known along the edge, it is possible to recover $\mathbf{n}_1(s)$ even when $\mathbf{n}_1$ unknown at initial point. Use Newton's method to compute $\mathbf{n}_1$ at initial point. This can be done because we now have two constraints on $\mathbf{n}_1$ viz. $E_1 = R(\mathbf{n}_1)$ and $[\mathbf{n}_1\,\mathbf{n}_2\,\mathbf{N}_c] = \mathbf{0}$. As these constraints are nonlinear, there is no guarantee of a unique solution. However it can be proved that for a reflectance map arising from a lambertian surface illuminated with a point light source, there can only be one or two solutions. We use the iterative scheme in (11) above to propagate each of the solutions, and pick the one which corresponds to the smaller error as measured by the functional in (2).

9 Shape constraints at a vertex

At a vertex we have (a) shading constraints from the brightness values on the faces which meet at the vertex, and (b) line drawing constraints from the projections of the edges incident at a vertex. Under certain conditions, there are enough constraints to uniquely determine the surface normals on each of the faces purely from the local information. We assume that the line labels of the edges meeting at the vertex have been determined earlier.

Consider vertex A in Figure 5 and enumerate the constraints on $\mathbf{n}_1$, $\mathbf{n}_2$, and $\mathbf{n}_3$–the surface normals to the three faces f_1, f_2, f_3 which meet at the vertex:

$$\begin{array}{ccc} E_1 - R(\mathbf{n}_1) = 0 & E_2 - R(\mathbf{n}_2) = 0 & E_3 - R(\mathbf{n}_3) = 0 \\ [\mathbf{n}_1\,\mathbf{n}_2\,\mathbf{N}_{12}] = 0 & [\mathbf{n}_2\,\mathbf{n}_3\,\mathbf{N}_{23}] = 0 & [\mathbf{n}_3\,\mathbf{n}_1\,\mathbf{N}_{31}] = 0 \\ \mathbf{n}_1^2 = 1 & \mathbf{n}_2^2 = 1 & \mathbf{n}_3^2 = 1 \end{array}$$

There are as many independent equations as unknowns. However as the equations are nonlinear, we are not guaranteed a unique solution. We can however put an upper bound on the number of solutions for typical reflectance maps. e.g. for $R(\mathbf{n}) = \mathbf{n}.\mathbf{s}$, there can be a maximum of 8 solutions [2]. Note that so far we have only used the information that each

[2]This result is true for any reflectance map whose level sets in p, q-space are conic sections

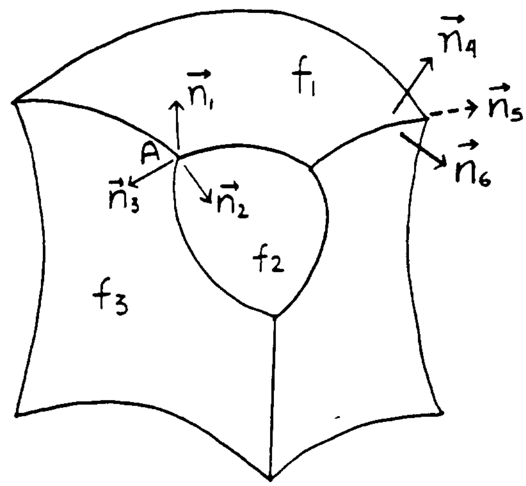


Figure 5: Shape recovery at a vertex

of the edges is connect (either convex or concave), but in fact we also know whether it is convex or concave. Pruning using this additional knowledge typically leaves only one solution.

For numerical computation, it is convenient to reduce the number of parameters by transforming to p, q-space and then use Newton's method to iteratively solve the system of nonlinear equations. A good initial guess can be obtained as follows: Assume that at each edge, the intersecting faces are mutually orthogonal. One can then determine the three surface normals in closed form. The solution/s which correspond to the known line labels at the vertex is/are used as starting values for Newton's method.

It can be seen that the method described above works for any vertex with $n \geq 3$ faces if and only if they are all visible. Alternatively, all the edges are either convex or concave. If any of the faces at the vertex is hidden, the constraint due to the brightness value on that face is missing, and there are fewer equations than unknowns.

However if there is exactly one hidden face, and the surface normal on one of the faces at the vertex is known (perhaps by propagation of information on one of the faces/edges that it lies on), then the number of independent equations again becomes equal to the number of unknowns and one can solve for the rest of the surface normals. In Figure 5, once $\mathbf{n}_4$ is known, $\mathbf{n}_5$ and $\mathbf{n}_6$ can be determined.

10 Shape Recovery given the Sparse Labelling

In sections 7, 8 and 9 we studied the constraint sets for image areas, curves and junctions, and developed numerical schemes for local shape recovery. For completeness, let us name these schemes and list them here:

1. *solve_face*: If $\mathbf{n}$ is known on a significant fraction of the boundary of an area, $\mathbf{n}$ can be computed in the interior of the area.

2. *solve_limb*: If an image curve is known to be the projection of a limb, then $\mathbf{n}$ can be determined along the limb.

3. *solve_connect_edge1*: $\mathbf{n}_1$, $\mathbf{n}_2$ known at initial point. Schemes (11) and (12) enable the computation of $\mathbf{n}_1(s)$ and $\mathbf{n}_2(s)$ along the edge, until either a vertex is reached or the edge becomes an occluding edge e.g. in Figure 4.

4. *solve_connect_edge2*: If $\mathbf{n}_2(s)$ is already known along the edge, then $\mathbf{n}_1(s)$ can be computed along the edge even if the initial value of $\mathbf{n}_1$ is not known. (This is as explained in the last paragraph of Section 8)

5. *solve_all_visible_vertex*: If all incident faces at a vertex with $k \geq 3$ faces are visible, then $\mathbf{n}_1, \ldots, \mathbf{n}_k$ can be computed on the k faces at the vertex.

6. *solve_almost_all_visible_vertex*: If only one of the faces at a vertex with $k \geq 3$ faces is hidden, then if any of $\mathbf{n}_1, \ldots, \mathbf{n}_k$ is known , the rest can be computed.

Of these only *solve_limb* and *solve_all_visible_vertex* are 'self starting'. The others need to make use of results of previous computations. For example, *solve_face* needs $\mathbf{n}$ on its boundary, computed by either *solve_limb* or one of the *solve_connect_edge* procedures. In turn, *solve_connect_edge_1* needs initial value information computed by one of the vertex solution procedures.

We now illustrate the process of global shape recovery with some examples. The hatched parts of the figures denote what has been freshly computed in a particular iteration. In figure 6, the computation starts with

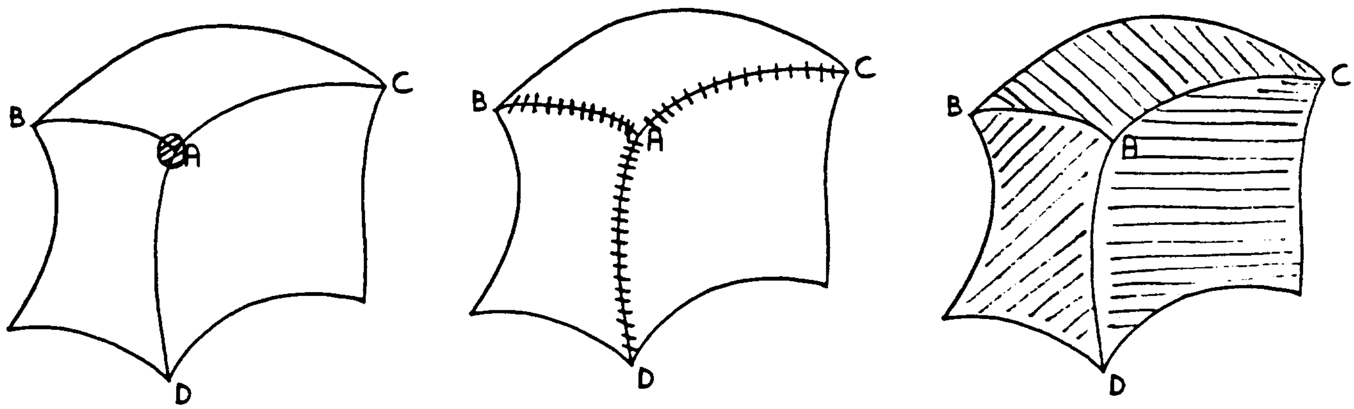


Figure 6: Sequence of steps in global shape recovery

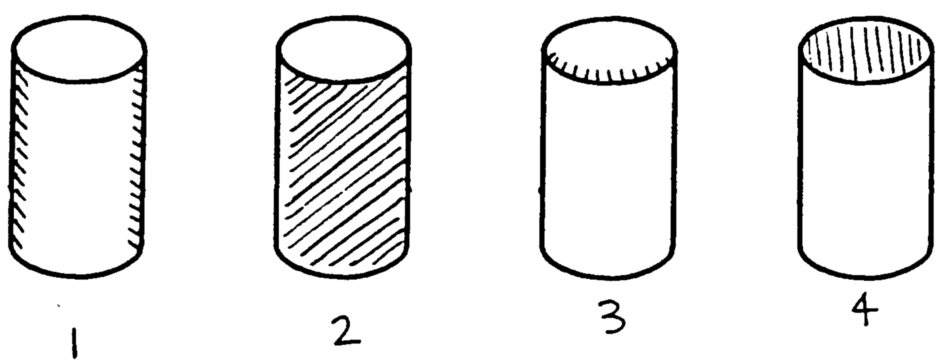


Figure 7: Sequence of steps in global shape recovery

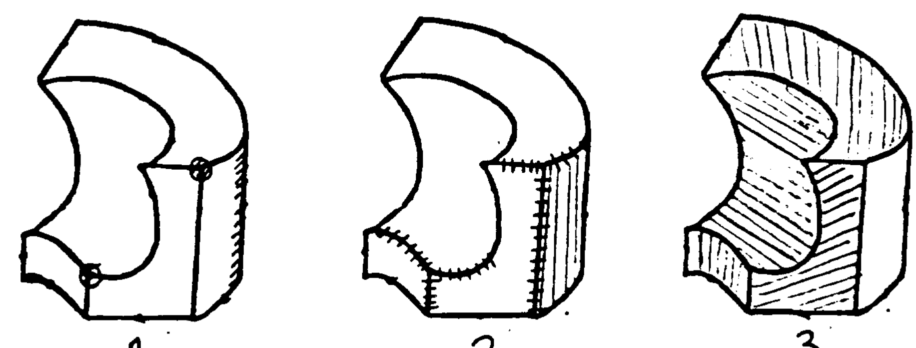


Figure 8: Sequence of steps in global shape recovery

determination of the surface normals to the three faces at A, followed by the computation of the surface normals along AB, AC and AD. Finally *solve_face* is applied to each of the three faces. Figures 7 and 8 are self-explanatory.

In the examples just considered, we started from the correct labelling. What is actually available to us are a set of sparse labellings e.g. Figure 2 among which only one is correct[3]. When we start from an incorrect sparse labelling, the 3–D shape computed will be incorrect and hence will not satisfy the line drawing and shading constraints within the margin of error permitted by the inaccuracies in the data (i.e. noise in pixel brightness values, errors in estimates of edge direction etc.). This gives us a way to prune away the shapes resulting from incorrect labellings. We now have the following scheme:

algorithm global_shape_recovery_PCS

1. Find applicable local shape recovery procedures.

2. Execute in parallel.

3. Check result (computed **n**) for consistency with image data within *margin-of-error*. If failure, terminate 'Wrong labelling'.

[3]In the sense that it corresponds to the particular scene being imaged. They are all consistent with the line drawing information.

4. If two or more local shape recovery procedures have independently computed $\mathbf{n}$ for the same location, check for consistency within *margin-of-error*. If failure, terminate 'Wrong labelling'.

5. If $\mathbf{n}$ known everywhere, or no more applicable procedures, terminate 'Solution'. Else go to 1.

The naive way to use this procedure is to run it , one labelling at a time. We can actually do much better. In Figure 2, and in the more extensive list of examples in [16], one notices that usually different labellings correspond to the same labels for inner curves–the outer curves have different labels corresponding to objects floating in air, resting on some support surface etc. To exploit this, the possible sparse labellings of a line drawing can be organized into a 'label inheritance tree', with the root corresponding to the set of curves which have the same labels, and branches corresponding to choices for curves which have multiple labelling possibilities. A least commitment strategy can then be used to avoid pursuing futile paths.

We have been somewhat glib in using the phrase *margin-of-error* in the algorithm for global shape recovery. Its exact specification is a difficult problem which we have not yet resolved to our satisfaction. There are two components of the error:

1. Error due to intrinsic noise in the image data. Consider the functional (1) in Section 5 without the regularization term. Even if we had determined the correct $\mathbf{n}(x, y)$, this functional would be non-zero. However we can come up with reasonable estimates. The first term would be equal to $\lambda_1 \sigma^2 A_I$ where σ^2 is the variance of the noise in the brightness values and A_I is the area of the image. Similar estimates can be made for the other terms.

2. Error accumulated during the numerical computation. This is of particular concern for the two *solve_connect_edge* procedures. In order to reduce this effect, the schemes in section 8 were chosen with stability as the primary concern. We need to estimate the magnitude of the accumulated error at the end point. We have been able to come up with upper bounds, which are however unduly pessimistic. We hope to have better luck with numerical simulations.

We are in the process of developing a computer implementation of the scheme proposed in this paper. There are several potential improvements that we are considering–solve at the edges as a boundary value problem rather than as an initial value problem, use a global minimization scheme once a crude shape has been obtained, use multi–grid methods to speed things up and help propagate global information etc.

To conclude, in this paper we have presented a coherent framework for representing the line drawing and shading constraints in a single image of curved objects. We also propose a scheme for computing the shape that satisfies these constraints. While the scheme needs further testing and development, it does demonstrate the inferential adequacy of our representation.

Acknowledgements

I thank W. Kahan for his expert advice on numerical analysis. This work was partially supported by an IBM Faculty Development award.

References

[1] H.G. Barrow and J.M. Tenenbaum, "Interpreting line drawings as three-dimensional surfaces," *Artificial Intelligence* **17**, pp. 75-116, Aug. 1981.

[2] J.M. Brady and A. Yuille, "An Extremum Principle for Shape from Contour," *Proceedings of IJCAI-8*, Karlsruhe, Aug. 1983, pp. 969-972.

[3] M.J. Brooks and B.K.P. Horn, "Shape and Source from Shading," A.I. Memo 820, Artificial Intelligence Laboratory, M.I.T. 1985

[4] A.R. Bruss, "Is what you see what you get," *Proceedings of IJCAI-8*, Karlsruhe, Aug. 1983, pp. 1053-1056.

[5] M.B. Clowes, "On seeing things," *Artificial Intelligence* **2** ,pp. 79-116, 1971.

[6] S. Geman and D.Geman, "Stochastic Relaxation, Gibbs Distributions, and the Bayesian Restoration of images," *IEEE Trans. on Pattern Analysis and Machine Intelligence,"* **6**, pp. 721-741, Nov. 1984.

[7] J.J. Hopfield, "Neural networks and physical systems with emergent collective computational abilities," *Proc. Natl. Acad. Sci. USA*, **79**, pp. 2554-2558, 1982.

[8] B.K.P. Horn and M.J. Brooks, "The variational approach to Shape from Shading," A.I. Memo 813, Artificial Intelligence Laboratory, M.I.T. 1985

[9] B.K.P. Horn, "Shape from Shading: A method for obtaining the shape of a smooth opaque object from one view," M.I.T Project MAC Rept. TR-79, 1970

[10] D.A. Huffman, "Impossible objects as nonsense sentences," *Machine Intelligence* **6** ,pp. 295-323, 1971.

[11] K.Ikeuchi and B.K.P. Horn, "Numerical Shape from Shading and Occluding Boundaries," *Artificial Intelligence* **17**, pp. 141-185, Aug. 1981.

[12] S. Kirkpatrick, C.D. Gelatt, M.P. Vecchi, "Optimization by simulated annealing," *Science* **220**, pp. 671-680, 1983.

[13] J.J.Koenderink," What does the occluding contour tell us about solid shape," *Perception* **13** ,pp. 321-330, 1984

[14] A.K.Mackworth, "Interpreting pictures of polyhedral scenes," *Artificial Intelligence* **4** ,pp. 121-137, 1973.

[15] J.Malik, "Labelling Line Drawings of Curved Objects," *Proceedings of the ARPA Image Understanding Workshop*, Miami, 1985, pp. 209-218.

[16] J.Malik, "Interpreting Line Drawings of Curved Objects," Tech. Rept. STAN-CS-86-1099, Computer Science Dept., Stanford University, 1985.

[17] J.Malik, "Interpreting Line Drawings of Curved Objects," *International Journal of Computer Vision* **1**, Jan. 1987.

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

[19] R.S.Millman and G.D. Parker, *Elements of Differential Geometry* New Jersey: Prentice-Hall Inc., 1977.

[20] T.Poggio, V.Torre, and C.Koch, "Computational vision and regularization theory," *Nature*, **317**(26), Sept. 1985.

[21] K.A. Stevens, "The visual interpretation of surface contours," *Artificial Intelligence* **17** , pp. 47-73, Aug. 1981.

[22] K.Sugihara, "Mathematical Structure of Line Drawings of Polyhedra – Towards Man–Machine Communication by means of Line Drawings," *IEEE Trans. on Pattern Analysis and Machine Intelligence* **4**(5),pp. 458-469, 1982.

[23] K.Sugihara, "An algebraic approach to Shape–from–Image Problems," *Artificial Intelligence* **23** , pp. 59-95, May 1984.

[24] A.N.Tikhonov and V.Y.Arsenin, *Solutions to Ill-posed Problems*, Winston, Washington, 1977.

[25] K.J.Turner,"Computer perception of curved objects using a television camera." PhD thesis, Edinburgh University, 1974.

[26] D.Waltz, "Understanding line drawings of scenes with shadows,"in *The Psychology of Computer Vision,* Ed. P.H. Winston ,McGraw-Hill, 1975.

[27] A. Witkin, D. Terzopoulos, and M. Kass, " Signal matching through scale space," *Proceedings of AAAI-5*, Philadelphia, Aug. 1986, pp. 714-719.