CVGIP: IMAGE UNDERSTANDING
Vol. 54, No. 3, November, pp. 349–357, 1991

Hypothesizing and Testing Geometric Properties of Image Data

KENICHI KANATANI

Department of Computer Science, Gunma University, Kiryu, Gunma 376, Japan

Received April 27, 1989; accepted April 12, 1991

A general formulation for detecting geometric configurations of inaccurate image data is presented. The basic principle is *hypothesizing* and *testing:* We first estimate an ideal geometric configuration that supposedly exists, and then check *to what extent the original edge data must be displaced* in order to support the hypothesis. All types of tests are reduced to computing a single *measure of edge displacement,* which provides a universal *measure of confidence* applicable to all types of decision-making. All the procedures are described by explicit algebraic expressions in N-vectors from the viewpoint of *computational projective geometry.* © 1991 Academic Press, Inc.

1. INTRODUCTION

In many computer vision problems, the 3-D inference of images is made through two stages: the *image processing stage,* extracting geometric primitives such as edges and feature points, and the *computational stage,* computing such 3-D properties as the shape and the position of the object. The accuracy of 3-D information computed in the second stage depends largely on the accuracy of the first stage. However, there is one crucial difficulty that cannot be resolved by simply increasing the accuracy—the issue of *consistency* of the image data.

In many problems, a clue to 3-D inference is provided by the information that image data, such as points and edges, have a special geometric configuration. For example, if several edges are concurrent (i.e., meeting at a common intersection when extended), we can infer that these edges may be parallel in the scene [2, 6]. However, the image data are obtained by pixel-based image processing, and inaccuracy due to noise and digitization is inevitable. As a result, the required consistency is usually violated for real image data.

The problem we consider in this paper is the question of how we can detect geometric configurations of image data that observed image data apparently *do not* possess. In the past, many researchers have treated this problem with ad hoc heuristics [1, 2, 6, 9, 10]. For example, three points are judged to be collinear if the angle defined by the two segments connecting these points is close to 180° within an arbitrarily set threshold value, or three lines are judged to be concurrent if the maximum separation among the three intersections is smaller than an arbitrarily set threshold value.

This type of ad hoc treatment requires introduction of measures to be thresholded and adjustment of the threshold values, but usually there is no justification for choosing particular measures and particular threshold values. In this paper, we will present a theory based on the *hierarchical structure* of image processing. Consider the following image processing scenario [7]:

1. From a gray-level image, edges are detected by application of an edge operator followed by thresholding and thinning.
2. Straight line segments are fitted to edges consisting of pixels that are nearly collinear.
3. Nearly collinear segments are grouped together and replaced by a single straight line.
4. Vertex positions are computed as intersections of these lines. If more than two lines approximately meet at a single point, an appropriate average is taken.
5. 3-D inferences are made from concurrency of lines and collinearity of points in the line drawing thus obtained.

If we look at these processes carefully, we realize that there is a *hierarchy* of data. First, we obtain *primary data* by pixel-based image processing techniques, then *secondary data* by numerical computation over the primary data. It is reasonable to assume that *primary data consist of edges,* because edge detection is usually the first step of image processing; all subsequent data are derived from edges.

Our strategy is as follows. If we want to detect some geometric configuration on the image plane, we first *hypothesize* the configuration that supposedly exists, and then test *how much the original edges must be displaced* in order to support the hypothesis. The hypothesis is accepted if the amount of displacement is smaller than a fixed threshold value, and rejected otherwise.

Although the threshold value must be empirically adjusted, the quantity to be thresholded is the amount of edge displacement alone. Since all image data are computed from the image data of lower levels, the computation can eventually be traced down to edges. This reduction to a *single criterion* serves as the universal *confidence level* of the decision, which enables us to compare the confidence levels of different types of decision with each other.

In the following, all expressions are given in terms of *N-vectors* representing points and lines in the image [4] so that all computation is always kept within a finite domain. Also, various approximation schemes are introduced so that all computation is reduced to evaluation of *explicitly* written expressions without requiring searches and iterations.

2. MATHEMATICAL PRELIMINARIES

Given an image, define an xy coordinate system on it such that the origin o *supposedly* indicates the position of the optical axis of the camera from which the image was obtained. Define an XYZ coordinate system such that the Z-axis passes through the image origin o perpendicularly. The X- and Y-axes are taken to be parallel to the image x- and y-axes, respectively, and the origin O is taken to be in distance f from the image origin o, where f is *supposedly* the distance between the center of the lens and the surface of the film (called the *focal length*) measured in the scale of the image coordinates (i.e., *pixels*) (Fig. 1).

Thus, we are viewing a given image as *perspective projection* of the 3-D scene by identifying the image as the *image plane*. However, this camera model *need not correspond to the true camera from which the image was obtained*. In other words, the camera model can be *hypothetical*. However, although the choice of the camera model does not essentially affect the following arguments, we assume that the focal length f is approximately equal to the true value (at least having the same order).

The unit vector starting from the viewpoint O and pointing toward a point P in the image is called the *N-vector* of the point P, and the unit vector normal to the plane defined by the viewpoint O and a line l in the image is called the *N-vector* of the line l [4] (see Fig. 1). From this definition, the N-vector of point (a, b) is[1]

$$\mathbf{m} = N\left[\begin{pmatrix} a \\ b \\ f \end{pmatrix}\right], \tag{2.1}$$

and the N-vector of line $Ax + By + C = 0$ is

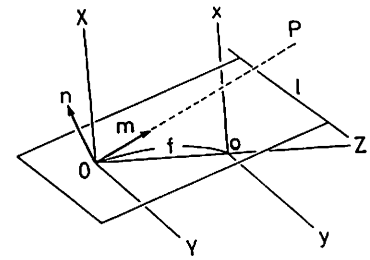

FIG. 1. N-vectors representing a point and a line on the image plane.

$$\mathbf{n} = \pm N\left[\begin{pmatrix} A \\ B \\ C/f \end{pmatrix}\right], \tag{2.2}$$

(the sign is arbitrary). Representing points and lines by the N-vectors has the advantage that computation is always done within a finite domain when otherwise infinite quantities are involved (e.g., when computing the intersection of parallel lines).

Note that although the original image itself is localized within a finite range around the image origin o, secondary data (i.e., points and lines computed from primary data) can lie *anywhere* on the image plane which conceptually extends infinitely. Representing points on the image plane by unit vectors has been familiar to some researchers [5], and a general theory of describing all geometrical relationships involving points and lines in terms of N-vectors is called *computational projective geometry* [4].

LEMMA 1. *If* $\mathbf{m}$ *is the N-vector of a point P in the image, the vector* $\overrightarrow{OP}$ *starting from the viewpoint O and ending at P is given by*[2]

$$\overrightarrow{OP} = \frac{f\mathbf{m}}{(\mathbf{m}, \mathbf{k})}. \tag{2.3}$$

Proof. Since $\mathbf{m}$ is the unit vector along $\overrightarrow{OP}$, we can put $\overrightarrow{OP} = c\mathbf{m}$ and determine the constant c in such a way that $(\overrightarrow{OP}, \mathbf{k}) = f$ (Fig. 2). ■

LEMMA 2. *If* $\mathbf{n}$ *is the N-vector of a line l in the image, the unit vector* $\mathbf{u}$ *perpendicular to the line l in the image (Fig. 3) is given by*

$$\mathbf{u} = \frac{\mathbf{n} - (\mathbf{n}, \mathbf{k})\mathbf{k}}{\sqrt{1 - (\mathbf{n}, \mathbf{k})^2}}. \tag{2.4}$$

[1] In this paper, $N[\mathbf{u}] = \mathbf{u}/\|\mathbf{u}\|$ denotes the normalization of vector $\mathbf{u}$, where $\|\mathbf{u}\|$ is the norm of vector $\mathbf{u}$.

[2] $(\mathbf{a}, \mathbf{b})$ designates the inner product of vectors $\mathbf{a}$ and $\mathbf{b}$. Throughout this paper, we put $\mathbf{k} = (0, 0, 1)^T$, where T denotes transpose.

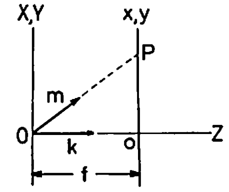

FIG. 2. Unit vector **m** lies along $\overrightarrow{OP}$, and $(\overrightarrow{OP}, \mathbf{k}) = f$.

Proof. We can see from Fig. 4 that vector $\mathbf{n} - (\mathbf{n}, \mathbf{k})\mathbf{k}$ is parallel to the image plane and perpendicular to line l. Since $\|\mathbf{n} - (\mathbf{n}, \mathbf{k})\mathbf{k}\|^2 = (\mathbf{n}, \mathbf{n}) - 2(\mathbf{n}, \mathbf{k})^2 + (\mathbf{n}, \mathbf{k})^2(\mathbf{k}, \mathbf{k}) = 1 - (\mathbf{n}, \mathbf{k})^2$, we obtain Eq. (2.4) after normalization. ■

LEMMA 3. *If* **n** *is the N-vector of a line l in the image, the unit vector* **v** *along the line l in the image (Fig. 3) is given by*

$$\mathbf{v} = \frac{\mathbf{n} \times \mathbf{k}}{\sqrt{1 - (\mathbf{n}, \mathbf{k})^2}}. \tag{2.5}$$

Proof. Since vector **v** lies in the plane defined by line l and the viewpoint O, and since the N-vector **n** is the unit surface normal to that plane, we have $(\mathbf{v}, \mathbf{n}) = 0$. Since it also lies in the image plane, we have $(\mathbf{v}, \mathbf{k}) = 0$. Thus, vector **v** is perpendicular to both **n** and **k**, and hence parallel to $\mathbf{n} \times \mathbf{k}$. Since $\|\mathbf{n} \times \mathbf{k}\| = \sqrt{1 - (\mathbf{n}, \mathbf{k})^2}$, we obtain Eq. (2.5) by normalizing $\mathbf{n} \times \mathbf{k}$. ■

LEMMA 4. *If* **m** *is the N-vector of a point P, and* **n** *the N-vector of a line l in the image, the distance h(P, l) of point P from line l (Fig. 3) is given by*

$$h(P, l) = \frac{f}{\sqrt{1 - (\mathbf{n}, \mathbf{k})^2}} \frac{|(\mathbf{m}, \mathbf{n})|}{(\mathbf{m}, \mathbf{k})}. \tag{2.6}$$

Proof. Take a point Q arbitrarily on line l (Fig. 3). The distance $h(P, l)$ is given by $|(\overrightarrow{QP}, \mathbf{u})|$, where **u** is the unit vector perpendicular to l. Noting that $\overrightarrow{QP} = \overrightarrow{OP} - \overrightarrow{OQ}$ and using Eqs. (2.3) and (2.4), we have

$$|(\overrightarrow{OP}, \mathbf{u})| = \frac{1}{\sqrt{1 - (\mathbf{n}, \mathbf{k})^2}} \left| \left(\frac{f\mathbf{m}}{(\mathbf{m}, \mathbf{k})} - \overrightarrow{OQ}, \mathbf{n} - (\mathbf{n}, \mathbf{k})\mathbf{k} \right) \right|. \tag{2.7}$$

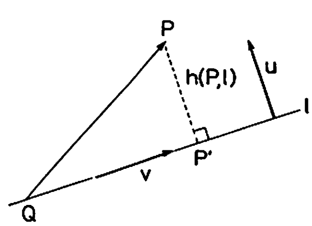

FIG. 3. Unit vector **u** is perpendicular to line l, and unit vector **v** lies along l. Point P' is the nearest point to point P on line l.

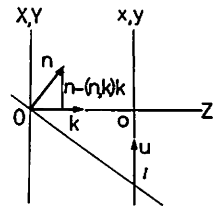

FIG. 4. Vector $\mathbf{n} - (\mathbf{n}, \mathbf{k})\mathbf{k}$ is parallel to the image plane and perpendicular to line l.

If we note that $(\overrightarrow{OQ}, \mathbf{n}) = 0$ and $(\overrightarrow{OQ}, \mathbf{k}) = f$, we obtain Eq. (2.6). ■

LEMMA 5. *If* **n** *is the N-vector of a line l, and* **m** *the N-vector of a point P in the image, the N-vector* **m**′ *of the point P′ nearest to point P on line l (Fig. 3) is given by*[3]

$$\mathbf{m}' = N\left[\mathbf{k} - (\mathbf{n}, \mathbf{k})\mathbf{n} + \frac{|\mathbf{m}, \mathbf{n}, \mathbf{k}|}{(\mathbf{m}, \mathbf{k})} \mathbf{n} \times \mathbf{k}\right]. \tag{2.8}$$

Proof. Define point P' by

$$\overrightarrow{OP'} = \frac{f}{1 - (\mathbf{n}, \mathbf{k})^2} \left(\mathbf{k} - (\mathbf{n}, \mathbf{k})\mathbf{n} + \frac{|\mathbf{m}, \mathbf{n}, \mathbf{k}|}{(\mathbf{m}, \mathbf{k})} \mathbf{n} \times \mathbf{k} \right). \tag{2.9}$$

It is easy to confirm that $(\overrightarrow{OP'}, \mathbf{k}) = f$. Hence, P' is a point on the image plane. It is also easy to confirm that $(\overrightarrow{OP'}, \mathbf{n}) = 0$. Hence, point P' lies on line l. It can also be shown that PP' is perpendicular to l, namely $(\overrightarrow{PP'}, \mathbf{v}) = 0$, where the unit vector **v** along l is given by Eq. (2.5). This is confirmed by substituting Eq. (2.5) and $\overrightarrow{PP'} = \overrightarrow{OP'} - \overrightarrow{OP}$, and noting that $\overrightarrow{OP} = f\mathbf{m}/(\mathbf{m}, \mathbf{k})$ by Eq. (2.3) and that $\overrightarrow{OP'}$ is given by Eq. (2.9). Thus, P' is the point on l closest to point P. Its N-vector is obtained by normalizing Eq. (2.9). ■

3. FITTING ALGORITHMS FOR INACCURATE DATA

We now consider an algorithm for detecting a "common intersection" of not necessarily concurrent lines. A reasonable method is as follows [3, 4]:

METHOD 1. *The N-vector* **m** *of the common intersection of lines* $l_1, \ldots, l_N$ *whose N-vectors are respectively* $\mathbf{n}_1, \ldots, \mathbf{n}_N$ *is estimated by the unit eigenvector for the smallest eigenvalue of the matrix*

[3] In this paper, $|\mathbf{a}, \mathbf{b}, \mathbf{c}| = (\mathbf{a} \times \mathbf{b}, \mathbf{c}) = (\mathbf{b} \times \mathbf{c}, \mathbf{a}) = (\mathbf{c} \times \mathbf{a}, \mathbf{b})$ denotes the scalar triple product of vectors **a**, **b**, and **c**.

$$\mathbf{N} = \sum_{\alpha=1}^{N} W_\alpha \mathbf{n}_\alpha \mathbf{n}_\alpha^{\mathrm{T}}, \tag{3.1}$$

where W_α is an appropriately chosen weight of line l_α.

Derivation. If point P of N-vector $\mathbf{m}$ is on all the lines, we must have $(\mathbf{n}_\alpha, \mathbf{m}) = 0$ for all $\alpha = 1, \ldots, N$ (see Fig. 1). Hence, the vector $\mathbf{m}$ is estimated by minimizing $\sum_{\alpha=1}^{N} W_\alpha (\mathbf{n}_\alpha, \mathbf{m})^2$ on the condition that $\mathbf{m}$ be a unit vector. This expression is rewritten as $\sum_{i,j=1}^{3} N_{ij} m_i m_j$, where the matrix $\mathbf{N} = (N_{ij})$, $i, j = 1, 2, 3$, is defined by Eq. (3.1). The minimum of this quadratic form in unit vector $\mathbf{m}$ is attained by the unit eigenvector of $\mathbf{N}$ for the smallest eigenvalue. ∎

Now, consider an algorithm for "fitting" a line to not necessarily collinear points. A reasonable method is as follows [3, 4]:

METHOD 2. *The N-vector* $\mathbf{n}$ *of the line fitted to points* $P_1, \ldots, P_N$ *whose N-vectors are respectively* $\mathbf{m}_1, \ldots, \mathbf{m}_N$ *is given by the unit eigenvector for the smallest eigenvalue of the matrix*

$$\mathbf{M} = \sum_{\alpha=1}^{N} W_\alpha \mathbf{m}_\alpha \mathbf{m}_\alpha^{\mathrm{T}}, \tag{3.2}$$

where W_α is an appropriately chosen weight of point P_α.

Derivation. If line l of the N-vector $\mathbf{n}$ passes through all the point, we must have $(\mathbf{m}_\alpha, \mathbf{n}) = 0$ for all $\alpha = 1, \ldots, N$ (see Fig. 1). Hence, the vector $\mathbf{n}$ is estimated by minimizing $\sum_{\alpha=1}^{N} W_\alpha (\mathbf{m}_\alpha, \mathbf{n})^2$ on the condition that $\mathbf{n}$ be a unit vector. This expression is rewritten as $\sum_{i,j=1}^{3} M_{ij} m_i m_j$, where the matrix $\mathbf{M} = (M_{ij})$, $i, j = 1, 2, 3$, is defined by Eq. (3.2). The minimum of this quadratic form in unit vector $\mathbf{n}$ is attained by the unit eigenvector of $\mathbf{M}$ for the smallest eigenvalue. ∎

The most widely known method for intersection estimation and line fitting may be the *least-squares method:* If $d(P, l)$ denotes the distance of point P from line l, the common intersection P of lines $l_1, \ldots, l_N$ is estimated by minimizing $\sum_{\alpha=1}^{N} W_\alpha d(P, l_\alpha)^2$, and the line l is fitted to points $P_1, \ldots, P_N$ so that $\sum_{\alpha=1}^{N} W_\alpha d(P_\alpha, l)^2$, is minimized. *This works well if all points and lines appear near the image origin.* However, if the points and lines are "computed data," they can exist anywhere (possibly at infinity), and the computation may break down. No computational problems arise in the above methods; all data are unit vectors, and no distinction is made whether the points and the line appear in the image or are computed to be at infinity.

4. MEASURE OF DEVIATION FOR PRIMARY LINES

In general, once an estimate is obtained, we must test "how good" the estimate is. As shown in the preceding section, we can treat points and lines in a symmetric way in terms of N-vectors. Indeed, the *duality* between points and lines is the essence of *projective geometry* [4]. However, this fundamental property is lost if we try to test the "quality" of estimation, because the roles of points and lines cannot be interchanged due to the hierarchy of image data: Edges are detected by an edge operator; lines are fitted to edges; points are defined as intersections of lines; new lines are defined by connecting them; . . . .

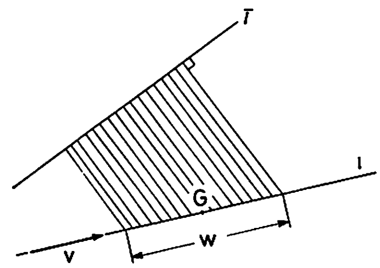

FIG. 5. The deviation of line l from line $\bar{l}$ is defined by the integral of the square distance from $\bar{l}$ along the primary segment of l.

Let us call lines fitted to edges *primary lines*. By definition, a primary line has a finite part that was fitted to an edge.[4] Let us call it the *primary segment*. Let w be its length. Let us call the midpoint G of the primary segment its *center*. Let m_G be the N-vector of the center G of the line l.[5]

Suppose we expect a line $\bar{l}$ with N-vector $\bar{\mathbf{n}}$ but instead observe a line l with N-vector $\mathbf{n}$.[6] How can we define the measure of deviation of the line l from line $\bar{l}$? Since it is very difficult to incorporate all factors, we adopt the following simplified model by noting that a longer line is generally more definite than a shorter one. Let $P(t)$ be a point on line l apart from the center G by distance t (the distance is signed appropriately), and $\mathbf{m}(t)$ its N-vector. As a measure of deviation of l from $\bar{l}$, consider the integral $\int_{-w/2}^{w/2} h(P(t), \bar{l})^2\, dt$, where $h(P(t), \bar{l})$ is the distance of point $P(t)$ from line $\bar{l}$ given by Eq. (2.6) (Fig. 5). From Lemma 3, we have

PROPOSITION 1.

$$\int_{-w/2}^{w/2} h(P(t), \bar{l})^2\, dt = \frac{w}{1 - (\bar{\mathbf{n}}, \mathbf{k})^2} \left(f^2 \frac{(\mathbf{m}_G, \bar{\mathbf{n}})^2}{(\mathbf{m}_G, \mathbf{k})^2} + \frac{w^2}{12} \frac{|\bar{\mathbf{n}}, \mathbf{n}, \mathbf{k}|^2}{1 - (\mathbf{n}, \mathbf{k})^2} \right). \tag{4.1}$$

[4] Each edge segment defines one primary line. Fitting a common line to multiple edges is considered later.

[5] Hence, the *data structure* for a primary line l consists of its N-vector $\mathbf{m}$, the N-vector $\mathbf{m}_G$ of its center, and the length w of its primary segment. The center and the length of the primary segment are defined only for primary lines; secondary lines are specified by their N-vectors alone.

[6] Throughout the rest of this paper, we use bars to denote quantities that are supposed to exist if no noise exists.

Proof. Since $\overrightarrow{OP}(t) = f\mathbf{m}(t)/(\mathbf{m}(t), \mathbf{k})$ by Lemma 1, we have from Lemma 4

$$h(P(t), \bar{l}) = \frac{|(\overrightarrow{OP}(t), \bar{\mathbf{n}})|}{\sqrt{1 - (\bar{\mathbf{n}}, \mathbf{k})^2}}. \tag{4.2}$$

Now, $\overrightarrow{OP}(t) = \overrightarrow{OG} + t\bar{\mathbf{v}}$, where $\bar{\mathbf{v}}$ is the unit vector along line $\bar{l}$. From Lemmas 1 and 3, we have

$$\overrightarrow{OP}(t) = \frac{f\mathbf{m}_G}{(\mathbf{m}_G, \mathbf{k})} + t\frac{\mathbf{n} \times \mathbf{k}}{\sqrt{1 - (\mathbf{n}, \mathbf{k})}}. \tag{4.3}$$

Substituting this in Eq. (4.2), and integrating its square over $-w/2 \le t \le w/2$, we obtain Eq. (4.1). ■

Now, we introduce an approximation. Since l is a primary line, its primary segment actually appears in the original image. This means that line l passes near the image origin (i.e., the distance from the image origin is small compared with the focal length f). Since we are considering a small deviation of line l, line $\bar{l}$ is also expected to pass near the image origin. Hence, their N-vectors $\mathbf{n}$ and $\bar{\mathbf{n}}$ are nearly parallel to the image plane. Thus, we have $(\mathbf{n}, \mathbf{k}) \approx 0$, and $(\bar{\mathbf{n}}, \mathbf{k}) \approx 0$. Since both $\mathbf{n}$ and $\bar{\mathbf{n}}$ are nearly parallel to the image plane, vector $\bar{\mathbf{n}} \times \mathbf{n}$ is nearly perpendicular to the image plane, and we have $|\bar{\mathbf{n}}, \mathbf{n}, \mathbf{k}| = (\bar{\mathbf{n}} \times \mathbf{n}, \mathbf{k}) \approx \|\bar{\mathbf{n}} \times \mathbf{n}\| = \sqrt{1 - (\bar{\mathbf{n}}, \mathbf{n})^2}$. Since the primary segment of line l actually appears in the original image, its center G is fairly close to the image origin. Hence, its N-vector $\mathbf{m}_G$ is nearly perpendicular to the image plane, and we have $(\mathbf{m}_G, \mathbf{k}) \approx 1$.

Substituting these into Eq. (4.1), we obtain

$$\int_{-w/2}^{w/2} h(P(t), \bar{l})^2\, dt \approx w\left(f^2(\mathbf{m}_G, \bar{\mathbf{n}})^2 + \frac{w^2}{12}(1 - (\mathbf{n}, \bar{\mathbf{n}})^2)\right). \tag{4.4}$$

We define the *measure of deviation* of line l from line $\bar{l}$ by

$$D(l, \bar{l}) = Cw\left(f^2(\mathbf{m}_G, \bar{\mathbf{n}})^2 + \frac{w^2}{12}(1 - (\mathbf{n}, \bar{\mathbf{n}})^2)\right), \tag{4.5}$$

where C is a factor reflecting the edge intensity and the edge width of the primary segment; let us call it the *edge strength factor*. The first term of the right-hand side can be regarded as measuring the "separation" of the center G of line l from line $\bar{l}$, while the second term can be regarded as measuring the "difference of orientations" of lines l and $\bar{l}$. A "physical analogy" helps our interpretation of this measure: If we imagine a hypothetical elastic "membrane" stretched between the primary segment of line l and the corresponding part of line $\bar{l}$ with a sliding rim along the line $\bar{l}$, Eq. (4.5) is viewed as the "energy" of the membrane, where the edge strength factor plays the role of the "elasticity constant."

5. COLLINEARITY TEST FOR PRIMARY LINES

Given multiple primary lines, how can we decide whether or not these lines can be judged as identical? This problem is very important in many low-level vision problems, because a straight object boundary is often detected as multiple fragmented edges in the presence of noise. In the past, this problem of "edge grouping" has been treated in ad hoc ways [1, 9].

Consider, for example, the two edges shown in Fig. 6(a). We can measure the distance δ and the angle α of deviation for each consecutive two edges, and judge the two edges as collinear if δ and α are respectively smaller than appropriately set threshold values. However, this may cause inconsistencies if we want to make a judgment about three or more edges unless additional heuristics are introduced (Fig. 6(b)). Thus, more and more ad hoc criteria become necessary as the problem becomes more and more complex.

Here, we take a simple and consistent approach. Consider the case of Fig. 6(a). First, we *hypothesize* that the two line segments are collinear, and *estimate* a line $\bar{l}$ that supposedly contains the two edges. Then, we *test* this hypothesis by computing *to what extent the original edges must be displaced if they all lie on the line l.* The hypothesis is accepted if the deviation is smaller than a threshold value, and rejected otherwise.

The case of Fig. 6(b) can be treated in the same way; we first hypothesize a line that supposedly contains all the edges by the *same* fitting scheme, and then test this hypothesis by computing to what extent these edges must be displaced, using the *same* measure of deviation. This measure is thresholded by the *same* threshold value. Thus, we need not introduce any new criteria, measures, or threshold values. Although we must introduce the measure of deviation and adjust the threshold value, they can be fixed for all types of problems.

A formal description is as follows. Let l_α, $\alpha = 1, \ldots, N$, be the primary lines to be tested for collinearity, and $\mathbf{n}_\alpha$, $\alpha = 1, \ldots, N$, their respective N-vectors. Let $\mathbf{m}_{G_\alpha}$, $\alpha = 1, \ldots, N$, be the N-vectors of their

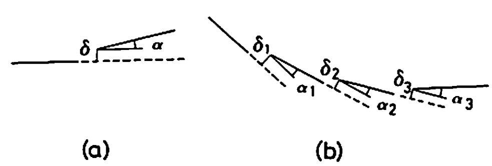

FIG. 6. (a) Two edges are judged as collinear if the angle α and the distance δ are separately smaller than appropriately set threshold values. (b) A different heuristic is necessary if we want to make a judgment about three or more edges.

centers G_α. The first step is hypothesizing a line $\bar{l}$. Here, we fit a line to the centers of the lines l_α by an appropriate method (e.g., Method 2 of Section 3 with each point weighted by $C_\alpha w_\alpha$, where C_α and w_α are the edge strength factor and the length of the primary segment, respectively, for the αth line). Let $\bar{\mathbf{n}}$ be the N-vector of the fitted line. The next step is testing this hypothesis. Using Eq. (4.6), we define the following measure of collinearity:

$$D(l_1, \ldots, l_N; \bar{l}) = \max_\alpha D(l_\alpha, \bar{l})$$

$$= \max_\alpha C_\alpha w_\alpha \left(f^2(\mathbf{m}_{G_\alpha}, \bar{\mathbf{n}})^2 + \frac{(w_\alpha)^2}{12}(1 - (\mathbf{n}_\alpha, \bar{\mathbf{n}})^2) \right). \quad (5.1)$$

The hypothesis is accepted if this value is below a fixed threshold value, and rejected otherwise.

6. CONCURRENCY TEST FOR PRIMARY LINES

Consider how to judge whether or not multiple primary lines are concurrent. This problem frequently arises in computer vision. For example, if we detect edges of an image of an object having corners, multiple (typically three) edges should precisely meet at one corner vertex. This, however, rarely happens for real images (Fig. 7(a)). Hence, we must make a judgment whether or not a given set of line segments define a single corner and, if so, estimate the corner position. Similarly, projections of edges parallel in the scene are concurrent, meeting, if extended, at a single *vanishing point* [2–6, 10]. However, we cannot expect this for real images (Fig. 7(b)). Hence, we must make a judgment as to whether or not a given set of line segments define a single vanishing point and, if so, estimate its position.

A naive way to solve this problem is to judge that lines

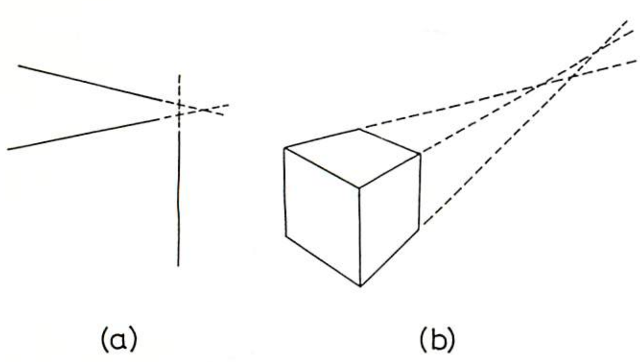

FIG. 7. (a) Edges that are supposed to meet at one corner vertex rarely define a single corner position. (b) Images of parallel line segments in the scene should meet, when extended, at a single vanishing point, but this cannot be expected for real images.

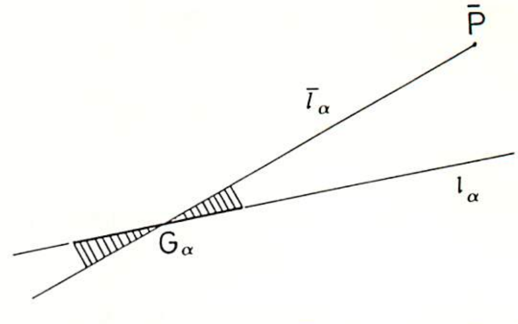

FIG. 8. Line $\bar{l}_\alpha$ passes through point $\bar{P}$ and the center G_α of line l_α.

are concurrent if the maximum separation of their intersections is below an appropriately set threshold value [2, 6]. However, the threshold value cannot be fixed. For example, tolerable errors must be severely restricted for the case of Fig. 7(a), while larger errors must be tolerated for the case of Fig. 7(b) because slight displacements of individual edges may cause a large deviation of their vanishing points.

As in the preceding section, our procedure is divided into two steps. First, we *hypothesize* concurrency and *estimate* the common intersection. Then, we *test* this hypothesis by computing *to what extent the original edges must be displaced to support the hypothesis*. The hypothesis is accepted if the deviation is below a fixed threshold value, and we can use the same threshold value for both Figs. 7(a) and 7(b).

Let l_α, $\alpha = 1, \ldots, N$, be the line to be tested for concurrency, and $\mathbf{n}_\alpha$, $\alpha = 1, \ldots, N$, their N-vectors. First, we hypothesize their common intersection $\bar{P}$ by computing its N-vector $\bar{\mathbf{m}}$ by an appropriate method (e.g., Method 1 of Section 3 with each line l_α weighted by $C_\alpha w_\alpha$, where C_α and w_α are the edge strength factor and the length of the primary segment, respectively, of the αth line). Note that point $\bar{P}$ can be located anywhere on the image plane (even at infinity), but the computation is always confined in a finite domain. The next step is to test this hypothesis. Draw a line $\bar{l}_\alpha$ passing through $\bar{P}$ and the center G_α of line l_α (Fig. 8). Let $\mathbf{m}_{G_\alpha}$ be the N-vector of the center G_α of line l_α. The measure of deviation, $D(l_\alpha, \bar{l}_\alpha)$, of line l_α from line $\bar{l}_\alpha$ is given as follows:

PROPOSITION 2.

$$D(l_\alpha, \bar{l}_\alpha) = \frac{C_\alpha}{12}(w_\alpha)^3 \left(1 - \frac{|\bar{\mathbf{m}}, \mathbf{n}_\alpha, \mathbf{m}_{G_\alpha}|^2}{1 - (\bar{\mathbf{m}}, \mathbf{m}_{G_\alpha})_2} \right). \quad (6.1)$$

Proof. First, note that a line of N-vector $\mathbf{n}$ passes through a point of N-vector $\mathbf{m}$ if and only if $(\mathbf{m}, \mathbf{n}) = 0$ (see Fig. 1). Also note that the N-vector $\mathbf{n}$ of the line passing through two points whose N-vectors are $\mathbf{m}_1$ and $\mathbf{m}_2$ is $N[\mathbf{m}_1 \times \mathbf{m}_2]$, because $\mathbf{n}$ must be orthogonal to both

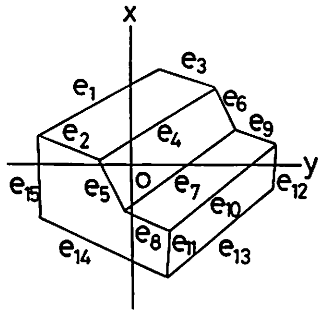

FIG. 9. A line drawing of a polyhedron.

of $\mathbf{m}_1$ and $\mathbf{m}_2$.[7] Let $\bar{\mathbf{n}}_\alpha$ be the N-vector of line $\bar{l}_\alpha$. Since line $\bar{l}_\alpha$ passes through the center G_α, we have $(\mathbf{m}_{G_\alpha}, \bar{\mathbf{n}}_\alpha) = 0$, and hence from Eq. (4.6)

$$D(l_\alpha, \bar{l}_\alpha) = \frac{C_\alpha}{12}(w_\alpha)^3(1 - (\mathbf{n}_\alpha, \bar{\mathbf{n}}_\alpha)^2). \tag{6.2}$$

The N-vector of the line $\bar{l}_\alpha$ passing through points $\bar{P}$ and G_α is given by

$$\bar{\mathbf{n}}_\alpha = N[\bar{\mathbf{m}} \times \mathbf{m}_{G_\alpha}] = \frac{\bar{\mathbf{m}} \times \mathbf{m}_{G_\alpha}}{\sqrt{1 - (\bar{\mathbf{m}}, \mathbf{m}_{G_\alpha})^2}}. \tag{6.3}$$

Substituting this in Eq. (6.2), we obtain Eq. (6.1). ■

Thus, we can test the concurrency by defining the following measure of concurrency:

$$D(l_1, \ldots, l_N; \bar{P}) = \max_\alpha D(l_\alpha, \bar{l}_\alpha)$$
$$= \max_\alpha \frac{C_\alpha}{12}(w_\alpha)^3\left(1 - \frac{|\bar{\mathbf{m}}, \mathbf{n}_\alpha, \mathbf{m}_{G_\alpha}|^2}{1 - (\bar{\mathbf{m}}, \mathbf{m}_{G_\alpha})^2}\right). \tag{6.4}$$

The hypothesis is accepted if this value is below a fixed threshold value.

Figure 9 is a line drawing based on a real image, where the edges labeled as shown.[8] If the above concurrency test is applied, three sets of parallel edges $\{e_1, e_4, e_7, e_{10}, e_{13}\}$, $\{e_2, e_3, e_8, e_9, e_{14}\}$, and $\{e_{11}, e_{12}, e_{15}\}$ are detected with a threshold value of $10^{-8}f^3$. The threshold value $10^{-8}f^3$ serves as the universal *confidence level* of this judgment (the less, the more confident). The same grouping is also obtained by the concurrency test algorithm of Kanatani [2, 3] based on spherical trigonometry, but his method employs many heuristics in an ad hoc way, and no clear-cut confidence value is given.

[7] These relationships are described in terms of N-vectors in a dual form with points and lines playing interchangeable roles. A formulation is presented in [4] under the name of *computational projective geometry*.

[8] The edge strength factor was taken to be unity.

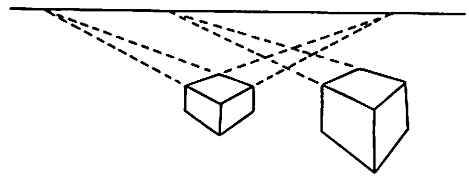

FIG. 10. The vanishing points of all the edges of horizontal faces must lie on a common "horizon."

7. COLLINEARITY TEST FOR SECONDARY POINTS

Consider how to make a judgment as to whether or not a given set of points is collinear. Recall that, in our model of image processing, points are always defined as intersections of lines. Let us consider secondary points, i.e., points defined as intersections of primary lines fitted to edges. The necessity of testing collinearity occurs in many problems of computer vision. For example, if horizontally placed object faces exist in the scene, all the vanishing points of their edges must all lie on a common *horizon* (Fig. 10). Hence, we can check whether individual faces are horizontal or not by testing collinearity.

However, exact concurrency cannot be expected for real data. Moreover, since points are computed data, they can be located anywhere on the image plane (even at infinity). The judgment must be done in a finite domain of computation, and the tolerance must depend on the positions and the configuration of these points. These issues are solved by the use of N-vectors and our strategy of hypothesizing and testing.

Let P_α, $\alpha = 1, \ldots, N$, be the points to be tested, and let $\mathbf{m}_\alpha$, $\alpha = 1, \ldots, N$, be their N-vectors. The first step is to hypothesize a line $\bar{l}$ that supposedly passes through these points. Let $\bar{\mathbf{n}}$ be the N-vector of the line fitted to these points by an appropriate method (e.g., Method 2 of Section 3). The next step is testing this hypothesis. Recall that each point is defined as the intersection of some lines. Let $l_\beta^{(\alpha)}$, $\beta = 1, \ldots, N_\alpha$, be the primary lines that define point P_α, and $\mathbf{n}_\beta^{(\alpha)}$, $\beta = 1, \ldots, N_\alpha$, their N-vectors. In other words, point P_α is given as the common intersection of primary lines $l_\beta^{(\alpha)}$, $\beta = 1, \ldots, N_\alpha$.

Let $\bar{P}_\alpha$ be the point on the hypothesized line $\bar{l}$ that are closest to point P_α. The N-vector $\bar{\mathbf{m}}_\alpha$ of point $\bar{P}_\alpha$ is computed by Eq. (2.8) of Lemma 5. Let $\mathbf{n}_\beta^{(\alpha)}$ be the N-vector of line $l_\beta^{(\alpha)}$, and $w_\beta^{(\alpha)}$ and $C_\beta^{(\alpha)}$ the length of its primary segment and the edge strength factor, respectively. Let $\mathbf{m}_{G_\beta^{(\alpha)}}$ be the N-vector of its center $G_\beta^{(\alpha)}$. We use the following measure to test collinearity (Fig. 11):

$$D(P_1, \ldots, P_N; \bar{l}) = \max_\alpha D(l_1^{(\alpha)}, \ldots, l_{N_\alpha}^{(\alpha)}; \bar{P}_\alpha)$$
$$= \max_\alpha \max_\beta \frac{C_\beta^{(\alpha)}}{12}(w_\beta^{(\alpha)})^3\left(1 - \frac{|\bar{\mathbf{m}}_\alpha, \mathbf{n}_\beta^{(\alpha)}, \mathbf{m}_{G_\beta^{(\alpha)}}|^2}{1 - (\bar{\mathbf{m}}_\alpha, \mathbf{m}_{G_\beta^{(\alpha)}})^2}\right). \tag{7.1}$$

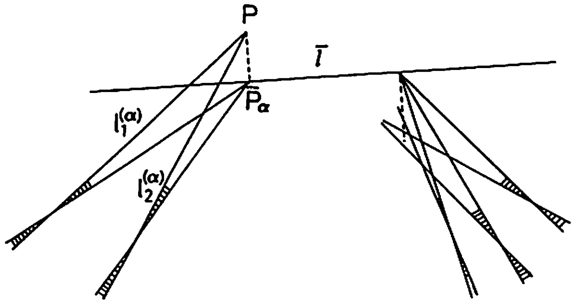

FIG. 11. The measure of collinearity of secondary points $P_1, \ldots, P_N$.

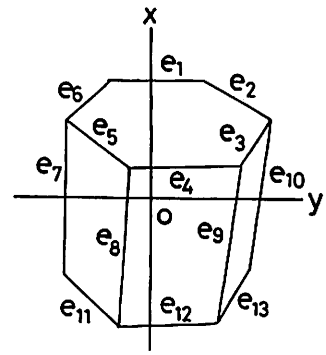

FIG. 12. A line drawing of a polyhedron.

Thus, we are measuring *to what extent the original edges must be displaced* so that the intersections all lie on the hypothesized line $\bar{l}$. The hypothesis is accepted if this measure is below a fixed threshold, and rejected otherwise.

Figure 12 is a line drawing based on a real image, where the edges are labeled as shown.[9] If the concurrency test of Section 6 is applied, four groups of concurrent edges, $\{e_1, e_4, e_{12}\}$, $\{e_2, e_5, e_{11}\}$, $\{e_3, e_6, e_{13}\}$, and $\{e_7, e_8, e_9, e_{10}\}$, are detected. If the collinearity test described above is applied to the vanishing points of the first three groups, they are judged to be concurrent with the threshold value of $10^{-8}f^3$, which serves as the universal *confidence level* of this judgment (the less, the more confident). This confidence level can be compared with the example in the preceding section, because both have the same meaning—the necessary displacements of original edges. Thus, we can assert that this judgment is *as confident* as that one. An algorithm of collinearity test is also suggested on the basis of spherical trigonometry in [2, 3], but the present approach is theoretically more consistent and useful.

8. CONCLUDING REMARKS

In this paper, we have presented a general formulation for testing geometric configurations of inaccurate image data by taking into account the *hierarchy* of image data resulting from image processing procedures. The basic principle is *hypothesizing* and *testing:* We first estimate an ideal geometric configuration that supposedly exists, and then check *to what extent the original edges must be displaced* in order to support the hypothesis. All types of tests are reduced to computing a single *measure of edge displacement*, which provides a universal *measure of confidence* applicable to all types of decision-making.

We have constructed, as typical examples of our strategy, the collinearity test for primary lines, the concurrency test for primary lines, and the collinearity test for secondary points. These tests are all very crucial to 3-D inference of images in computer vision. In our formulation, no explicit forms of probability distribution need be introduced. All the procedures are described by explicit algebraic expressions in N-vectors, so no searches or iterations are required.

The discipline proposed in this paper is consistent as compared with ad hoc/heuristic approaches adopted by many researchers in the past. It is also simple and easy to implement from a practical point of view. From a purely theoretical point of view, however, there remain the following issues to be settled:

1. Although the measure of edge displacement introduced in this paper is reasonable in view of its geometric meaning, the subsequent procedures would be the same if some other measures were adopted. This is not a problem from a pragmatic point of view, but as a "theory" it is not desirable: can we justify our measure by means of more fundamental principles?

2. In this paper, estimation and testing are treated separately in the sense that any estimation can be tested for its validity by our method. But can we derive an "optimal estimate" based on the hypothesis by a theoretical means? For example, if Methods 1 and 2 in Section 3 are to be used, can we find "optimal weights" W_α such that the resulting estimates have the highest reliability? From a practical viewpoint, all estimates to be tested should be computed easily, but from a theoretical viewpoint, such "optimal estimates" should be tested.

3. Although all types of tests are reduced to a single measure of edge displacement with a single threshold, this threshold, as well as the edge strength factor, must be empirically adjusted. This is certainly an advance as compared with adjusting problem-dependent thresholds each time; we only need to consider how images were processed. However, it would be preferable to have a theory to determine such image-based values.

[9] The edge strength factor was taken to be unity.

All these three issues can be settled affirmatively by means of *statistical inference:* If we introduce a *statistical model* of image data, i.e., the probability that particular image data are observed, we can construct *maximum likelihood estimators* [8], or we can test hypotheses by setting the threshold values so that the probability of misdetection is below some level, e.g., the 5% level. However, in order to derive such a theory, we need long mathematical preliminaries and intricate mathematical techniques. This will be discussed as separate papers in the future.

Our final remark is on the "goodness" of our approach. Take the edge grouping procedure discussed in Section 5, for instance. One may be tempted to ask how "good" it is as compared with other ad hoc/heuristic methods. Such a question is "ill-posed". For example, applying two edge grouping algorithms to a real image, one may say, "This edge and this edge belong to the same house wall, so they should be grouped together. Let's see. Algorithm A groups them together, while Algorithm B does not. So, algorithm A is better than algorithm B." This is a typical example of the subjective reasonings that have confused and misguided many computer vision researchers in the past. If we are constructing a *low-level* algorithm based on image properties alone, we may as well ask to what extent the output is affected by image noise or edge operators, but it does not make sense to compare the algorithm with an "oracle" ("They belong to the same house wall, so . . .", etc.). The procedure of Section 5, on the other hand, is based on a clearly stated logic on image properties. If it does not group two edges which belong to the same house wall, then that image is *as such*. This is all we can expect: we cannot expect such a low-level algorithm to be intelligent enough to recognize and identify objects in the scene. To emphasize the necessity of objective reasonings based on consistent and sound logic is also one of the purposes of this paper.

REFERENCES

1. K. M. Andress and A. C. Kak, Evidence accumulation & flow of control in a hierarchical spatial reasoning system, *AI Magazine* **9** (1988), 75–94.
2. K. Kanatani, Reconstruction of consistent shape from inconsistent data: Optimization of 2½D sketches, *Int. J. Comput. Vision* **3** (1989), 261–292.
3. K. Kanatani, *Group-Theoretical Methods in Image Understanding*, Springer-Verlag, Berlin, 1990.
4. K. Kanatani, Computational projective geometry. *CVGIP: Image Understanding*, **54** (1991), 333–348.
5. M. J. Magee and J. K. Aggarwal, Determining vanishing points from perspective images, *Comput. Vision Graphics Image Process.* **26** (1984), 256–267.
6. P. G. Mulgaonkar, L. G. Shapiro, and R. M. Haralick, Shape from perspective: A rule-based approach, *Comput. Vision Graphics Image Process.* **36** (1986), 298–320.
7. A. Rosenfeld and A. C. Kak, *Digital Picture Processing*, Vols. 1, 2, 2nd ed., Academic Press, Orlando, FL, 1982.
8. I. Weiss, Line fitting in a noisy image, *IEEE Trans. Pattern Anal. Mach. Intelligence* **PAMI-11** (1989), 325–329.
9. R. Weiss and M. Boldt, Geometric grouping applied to straight lines in *Proceedings, IEEE Conference on Computer Vision and Pattern Recognition, June 1986, Miami Beach, Florida*, pp. 656–661.
10. R. Weiss, H. Nakatani, and M. Riseman, An error analysis for surface orientation from vanishing points, *IEEE Trans. Pattern Anal. Mach. Intelligence* **PAMI-12** (1991), 1179–1185.