COMPUTER VISION AND IMAGE UNDERSTANDING
Vol. 68, No. 2, November, pp. 190–208, 1997
ARTICLE NO. IV970552

Detecting Perceptually Parallel Curves: Criteria and Force-Driven Optimization

Horace H. S. Ip and W. H. Wong

Image Computing Group, Department of Computer Science, City University of Hong Kong, Tat Chee Avenue, Kowloon, Hong Kong

Received July 22, 1994; accepted September 16, 1996

We have developed several theorems for the detection of parallel curves in the continuous space. In this paper, we studied issues in carrying the continuous algorithm to the discrete case and also the perceptual characteristics leading to human recognition of parallelism. By formulating these properties in terms of several distinctive forces, we developed a force-driven model as a new optimization strategy to perform correspondence establishment between points in the matching curves. This force-driven mechanism provides a good coupling (or correspondence matching) result, which is the prerequisite for the correct detection of parallelism between curves. Convergence of the algorithm and implementation efficiency are also investigated and discussed. Experimental results on the relative weightings of these forces also shed light on the perceptual priority imposed by the human vision system. © 1997 Academic Press

1. INTRODUCTION

Perceptual organization was first described by the Gestalt psychologists [26]. They illustrated that primitive grouping in our vision system is done on proximity, collinearity, curvilinearity, parallelism, symmetry, closure and repetitive patterns. Vision researchers believe that this fascinating capability can be applied to reduce the search space in object recognition, by limiting the number of possibilities and the order in which these possibilities are analyzed [13].

In this paper we propose a force-driven approach to the detection of parallelism between curves. We begin with an outline of related work in perceptual organization in Section 2 to place our work into perspective. In Section 3 we briefly review the algorithm for the detection of parallelism for continuous curves decribed in [33]. The issues introduced when we apply the algorithm to the discrete curves are also discussed. Then a force-driven coupling mechanism is presented in Section 4. The parallel detection algorithm for digital curves is portrayed in Section 5. Experimental results and discussions are addressed at the end of the paper.

2. TRADITIONAL APPROACHES

Witkin and Tenenbaum [29] first proposed applying perceptual organization to machine vision. They hypothesized the visual system as capable of *guessing what is important without knowing why;* hence perceptual organization was a primitive level of inference that could operate on its own without high-level knowledge concerning the scene. In their structure-based vision paradigm, the role of perceptual grouping was to act as a primitive inference mechanism that provides *semantic precursors* for higher level processing. Lowe also proposed applying perceptual grouping to reduce the search space in machine recognition and did some canonical work in formulating a computational model for perceptual organization [13]. Since then, detection of perceptually significant properties has been well studied by the vision community, although most of the work was restricted to straight line segments [13, 14, 16, 17]. In general, these nonaccidental properties have been widely accepted as clues to feature grouping [9], as they are very likely to be originated from a single object. A good feature grouping scheme removes numerous unessential combinations; hence it dramatically reduces the computational loads of high level vision problems like object recognition.

Recently, detection of these properties for curves has attracted the attention of many researchers. For example, Dolan and Weiss [4] applied perceptual organization to group segments into higher level curvilinear structures. Mohan and Nevatia [18] identified short and relatively straight edge segments from images and grouped them into larger curvilinear structures based on collinearity, continuity, and proximity. Exhaustive search was applied to these segments to check for possible parallel and symmetry rela-

1077-3142/97 $25.00

tions, which were modeled as a constraint satisfaction problem implemented using a Hopfield network. Sarkar and Boyer [22, 23] applied perceptual organization to the indication of geometrical features like circles, ellipses, parallelograms, and ribbons in an image. They attempted a hierarchical approach to obtain coherent global structures from local features through voting, graph theoretic enumeration, and knowledge-based reasoning. For each level of hierarchy, they split the work into preattentive and attentive grouping. Preattentive grouping aims at detecting various relations like parallelism, curved symmetry, closure, etc. [23]. Attentive grouping targets at providing hypotheses for parallelograms, ribbons, quadrilaterals with associated probabilities [22]. In [25], Ulupinar and Nevatia did provide a very simple scheme to describe parallelism for planar curves. However, as remarked in [33], their scheme is too loose for a reliable detection of parallelism.

A closely related research area is shape matching. Parallelism detection can be described as shape matching added with extra constraints like proximity and similarity in orientation. Almost all solutions to curve matching were based on descriptor distance measures. In general, local descriptors were favored as partial matching could not be handled if global descriptors were used. Most solutions did not use identifiable features for better robustness (for example, [3, 6, 15, 24]). Usually, transformation parameters were estimated and rotational invariance was achieved by a brute force circular shift on the descriptors. The resulting complexity is not promising. Huttenlocher and Ullman [8] put forth a much more efficient method by separating alignment from matching. Alignment was assisted by feature identification. Multiresolution strategy was applied to speed up processing. Rosin [21] extended the codons of Hoffman and Richards [7] and adopted the multiscale approach to construct a hierarchical curve representation for better model matching. This method is computationally expensive and implementation is difficult because of the increased number of the primitive codons and the rules governing their admixture.

The force-driven formulation proposed in this paper resembles the active contour models [10], which was a computational model based on the deformable template discussed in [27]. In Kass' formulation, variational calculus was applied to the energy minimization procedure. This requires the energy to be a differentiable function; hence, local geometry of the active contour models is difficult to constrain [1]. In addition, discretizing the formulation introduces numerical instability. Dynamic programming was proposed to overcome these problems [1]. The center of a square window is positioned at each snaxel, within which m candidate positions are evaluated. For an active contour made up of n snaxels, the global minimum could be found but the worst case complexity is of $O(m^3n)$. Determination of window size is another issue to be solved. A greedy approach has been adopted to reduce the complexity to $O(mn)$ [28]. However, global optimality is not guaranteed and the window size issue is not addressed.

At first sight, it may appear that the force-driven formulation given here is conceptually related to the active contour model. The similarity lies in that both are achieving global optimization through local optimization. However, we would like to emphasize here that the force-driven mechanism is *not* an active contour model, it is an optimization technique on its own. The mathematical formulations given in Section 4.2 make this distinction clear. It is worth emphasizing here that the minimization of the energy terms in the active contour model requires certain optimization techniques. In the original formulation [10], snake energy minimization was achieved by means of variational calculus. Other choices of optimization techniques are dynamic programming [1], minimax strategy [11], greedy algorithm [28], etc. Our force-driven model emulates criteria and constraints as competing forces to counteract on each other and alter the results. The optimal solution is obtained when either all these forces vanish or they cancel out each other.

Concerning the methods we reviewed earlier, some of them make use of high level knowledge and are quite contradictory to the underlying philosophy of perceptual organization. Others need the determination of transformation parameters to carry out the necessary transformation before the match. We believe this kind of time-consuming preprocessing is not required in the human visual system. Instead some sort of fuzziness is built to the human visual system to allow matching of moderately deformed shapes. It is this sort of fuzziness we want to emulate in our algorithm. That is our motivation to derive an algorithm which could detect *perceptually* parallel curves as well. Also, the high computational complexities of many of the discussed algorithms indeed ruled out the possibility of applying them as a precursor mechanism. In this paper, we attempted an analytic approach to the problem and introduced an optimization strategy which allows efficient parallelism detection without an explicit computation of shape descriptions.

In [33], we studied models of parallelism for curves and developed a computational framework for parallelism detection (a brief review is given in Section 3). We divided the problem into two parts: correspondence establishment and parallelism vertification. By breaking curve segments at points of zero-curvature or curvature extrema, we put forth a divide-and-conquer algorithm for parallelism detection. However, as we remarked at the end of that paper, in practice, detection of zero-curvature points and curvature extrema is no trivial task. Furthermore, accurate determination of tangent and curvature information from digitized curves is nearly impossible. Hence, instead of relying heavily on accurate curvature and tangent information, we present mechanisms for the basic algorithm to cope with, not

only the quantizing errors and noise in digital curves, but also the uncertainties associated with the underlying curvature estimators. The main contributions of this paper are:

- we enhanced the original parallel detection algorithm, which was formulated for continuous curves, to work for discrete curves,
- we relaxed the original constraints, which are true only under orthographic projections, to cater for deformations due to perspective projections and ultimately for detecting perceptual parallelism,
- to the end of perceptual parallelism detection, we studied the perceptual characteristics leading to a parallelism description,
- we addressed the correspondence problem explicitly and proposed a new force-driven optimization strategy to tackle the problem.

3. PARALLELISM DETECTION ALGORITHM FOR CURVES

In [33], we studied three classes of parallelism: translation parallelism, railroad parallelism, and central similarity transformation parallelism. For parallelism to exist between two curves, it is necessary to apply a coupling process to find for each point in one curve (the *active curve*) a corresponding point in the other (the *passive curve*) and vice versa, which conform to the following constraints:

1. the two points coupled (we named them *conjugate pair*) must have compatible tangent orientations;

2. a point of curvature extremum should be coupled to a point of curvature extremum;

3. a point with zero-curvature should be coupled to a point with zero-curvature.

A curve C_q is represented in the parametric form as $C_q(s) = (x_q(s), y_q(s))$, where s runs from 0 to curveLength(C_q). To facilitate parallelism detection, we represent tangent vectors as *direction-dependent tangent* vectors (DDT) [30]. Denote $\varphi_q(s)$ the tangent vectors at $C_q(s)$, valued in the range $[0, 2\pi)$, and denote $\psi_q(s)$ the corresponding DDT vector,

$$\psi_q(s) = \begin{cases} \varphi_q(s) + 2\pi & \text{if } \Gamma(\langle\varphi_q(s)\rangle, \langle\varphi_q(s+\varepsilon)\rangle) \\ \varphi_q(s) & \text{otherwise,} \end{cases} \tag{1}$$

where the relation $\Gamma(\langle a\rangle, \langle b\rangle)$ is defined as

$$\Gamma(\langle a\rangle, \langle b\rangle) \Leftrightarrow \begin{vmatrix} \cos(a) & \cos(b) \\ \sin(a) & \sin(b) \end{vmatrix} \geq 0 \tag{2}$$

to denote two unit vectors, with orientations a and b, are positively oriented. By defining an operator Left($\cdot$) for DDT's,

$$\text{Left}(\psi) = \begin{cases} \psi & \text{if } \psi < 2\pi, \\ (\psi - \pi) \bmod 2\pi & \text{if } \psi \geq 2\pi, \end{cases} \tag{3}$$

equivalency between two DDT's, denoted by $\psi_1 \triangleq \psi_2$, is defined as

$$\text{Left}(\psi_1) = \text{Left}(\psi_2) \Leftrightarrow \psi_1 \triangleq \psi_2. \tag{4}$$

DDT representation explicitly incorporates tangent orientation with respect to the direction of curve following; hence, concavity information is kept implicitly. It helps to prevent false alarms in parallelism detection [30]. Here we outlined the parallelism detection algorithm described in [33]; please refer to original paper for full details and related mathematical theorems.

Step 1. *Decomposition of curves into simple segments*

Step 2. *Prescanning for possible coupling between simple segments*

Step 3. *Detailed coupling and parallelism verification*

Step 4. *Parallelism reporting.*

The basic algorithm breaks curves into simple segments for efficient parallelism detection. A *simple segment* is a section of a curve in which there exists no zero-curvature point, except at both ends or all the points along the segment are of zero-curvature (i.e., a straight line). Formally,

C_x is a simple segment if either

$$\begin{cases} \kappa_x(s) \neq 0 & \text{for } 0 < s < \text{length}(C_x) \\ \kappa_x(s) = 0 & \text{for } 0 \leq s \leq \text{length}(C_x), \end{cases}$$

where $\kappa_x(s)$ represents the curvature at the point $C_x(s)$. Simple segments from two curves are tested for parallelism if and only if the DDT's at the segment endpoints are equivalent. We apply this constraint to speed up the detection process. In all discussions, we use C_i for the active curve and C_j for the passive curve. To reduce the need for symbols, we employ $C_i(s)$ and $C_j(f(s))$ to indicate the point $C_j(f(s))$ on the passive curve is coupled to the point $C_i(s)$ on the active curve.

3.1. *Addressing the Problems in the Discrete Case*

Applying the basic algorithm to the discrete case, we need to take care of two issues. The first issue is the number of points we should have coupled in step 3 (detailed coupling) before we say two segments are parallel. The idea of *all* points in the continuous case is definitely not applica-

ble for the discrete case, nor is it suitable for implementation. Sampling is required. The way we sample points from the two digitized curves for coupling verification is a trade-off between accuracy and efficiency. The larger number of points we check against, the greater is the computational need but, at the same time, the higher is the confidence we have on our results. Following the idea of using salient points to approximate curves [20], we choose to check against these perceptually salient points only. We use $C_i(S_p)$ to denote the salient points in curve C_i, the values of $p = 0, 1, 2, \ldots, \max_p$. Segments between salient points can usually be approximated by straight lines. Hence, it is not worthwhile to perform coupling for the points in between. An efficient solution is to apply the correspondence coupling step to these salient points only and the rest done by linear interpolation. Then we test and group the number of consecutive segments that can be described as parallel.

The second issue is the effect of noise and aliasing on curvatures and DDT vectors estimation. The approach discussed in [33] requires information about curve tangent and curvature. However, we lack a good digital tangent and curvature estimator for digital figures [34]. Accurate detection of zero curvature and curvature extrema is again difficult. Even with the improvements obtained from the conditioning algorithm we proposed in [31], the best we can obtain is an idea about the number and locations of these zero-curvature points. Hence, rather than relying on the stability of these underlying techniques, our force driven solution is formulated such that the algorithm is able to tolerate instability and errors associated with these schemes.

3.2. *Addressing the Problems in Perspective Projection*

The algorithm described above (and in greater detail in [33]) is able to detect parallelism for 3D continuous curves under orthographic projection to 2D images. Under perspective projection, which is common in the real world, parallel lines (hence also parallel curves) are no longer parallel in the strictest sense. However, under a wide range of viewing angles, we can still perceive parallelism in these curves. We want to enhance the algorithm to detect parallelism for curves under perspective projection.

The first implication is that the coupling strategy used in the original algorithm must be relaxed. Perspective projection is a fractional linear transformation [12]; hence, foreshortening is nonuniform. Parallel lines are no longer parallel after projection. The equivalent DDT criterion based on which we couple points in the original algorithm is no longer valid. Instead we need to grant a bound within which the DDT values can fluctuate during coupling.

The second implication is that we can no longer identify the class of parallelism as discussed in [33]. Needless to say, we do not have enough information to discriminate railroad parallelism from central similarity transformation parallelism or translational parallelism from the available data without making further inference using global information and object models.

These implications lead us to believe that the algorithm should be able to detect perceptually parallel curves which are common in line drawings, hand-drawn sketches, medical images, and natural scenes. The same algorithm could also serve as a backbone mechanism for contour-based stereo or motion techniques. However, in perceptual parallelism there are in fact no exact rules to follow; it is difficult to formulate perceptual parallelism in mathematical terms. Instead, we attempt to formulate this by inspecting the perceptual characteristics of parallel curves. This will be explained in greater depth in the next section. Besides, we can no longer apply the tests described in [33]. Instead, we establish new parallelism tests detailed in Section 5.3.

4. FORCE-DRIVEN CORRESPONDENCE MATCHING

In this section, first we review the criteria for parallelism detection under perspective projection and perceptual parallelism. They are the perceptual characteristics of parallel curves. Then we shall present our force-driven model.

4.1. *Perceptual Characteristics of Parallel Curves and Associated Criteria*

We identified four perceptual characteristics of parallel curves and associated with each of them a criterion. They are discussed one by one below.

Sleepers Criterion. From the parallelism models described in [33] we expect lines joining conjugate pair (we called these lines *sleepers*) to be either parallel to each other or concurrent (being met at a point). This gives rise to our first criterion: for a pair of parallel simple segments, salient points on C_i should be coupled to points on C_j such that the sleepers joining them are either parallel to those at the endpoints or concurrent with those at the endpoints. Notice here that the property of concurrent is invariant under perspective projection.

Tangent Alignment Criterion. For parallelism to exist, the tangents at conjugate pairs must agree. Although under perspective projection this requirement no longer holds, the deflection still must not be too large for parallelism to be observed by human. In the absence of any information on global structure about the curves, we can only make the assumption that tangents should agree for parallelism to occur.

Similarity Criterion. As described in Section 3.1, we approximate the curve C_i using the set of salient points $C_i(S_p)$. This criterion expects that the corresponding points

$C_j(f(S_p))$ also approximate the curve C_j if the two curves are parallel (or similar in shape).

Elasticity Criterion. For the pair of simple segments being coupled, distances between successive coupled points $C_j(f(S_p))$ are lengthened or shortened more or less uniformly during the coupling process.

4.2. *The Force-Driven Model*

At first sight, it seems that we have a distance minimization problem involving the four criteria listed above. In fact, this is not exactly the case in two respects. First, we have in addition two implicit constraints,

1. monotonic coupling order [25], i.e., sleepers must not intersect each other;

2. the coupling of inflection points to inflection points and curvature extrema to curvature extrema of the same concavity imposed by the divide-and-conquer algorithm described in Section 3 (see also [33]).

Second, the perceptual characteristics given in the previous section are not hard and fast rules that must be followed. Some of them are simplified assumptions made on the basis of the parallelism models studied in [33]. Consequently, they only *suggest* how the mapping scheme should be constructed. It is obvious that to find a mapping scheme satisfying all of them is impractical (and does not make much sense). This limits the applicability of the traditional optimization techniques. Our solution to the barricade is to formulate these criteria into a number of competing forces and let them counteract each other until they arrive at an equilibrium. Each criterion in question is represented by a force which is defined such that its magnitude is proportional to the amount of deviation from the expected result based on the associated criterion, while the direction of force is towards the most promising direction to reduce that deviation.

To put it in formal terms, constraints or criteria $Q_i(x)$ in the force-driven framework must be formulated as objective functions in the format

$$Q_i(x) \equiv q_i(x) = 0, \tag{5}$$

where x can be a vector and $q_i(x)$ representing the corresponding function for the constraint. The rationale behind such a composition is that when the constraint or criterion is satisfied, the function value of the corresponding objective function vanishes. In other words, nonzero function values signify deviation from the expectancy evaluated on that particular constraint or criterion. For each nonzero $q_i(x)$, we measure the gradient $g_i(x)$; then the two are combined to create a force vector $\tau_i(x)$, with direction $-g_i(x)$ and magnitude $w_i q_i(x)$, where w_i is the weight assigned to the objective function $q_i(x)$. The maximum possible number of force vectors acting at a point is the number of objective functions constructed for the problem. These force vectors are added together to give the net resultant force $\tau(x)$ for x. The direction indicated by $\tau(x)$ is the searching direction where hopefully a better solution can be found. Now, the distinction between the force-driven model and the active contour model is obvious. First, the active contour model does not require objective functions to follow the scheme given in (5). Second, the search direction in the force driven model is determined for each criterion (or constraint) separately, unlike the case for active contour model where minimization is done on the collection of energy terms as a whole. As can be seen from the formulations given in subsequent sections, this arrangement allows us to obtain the gradient direction in a very efficient manner.

Given two simple segments, one from each curve, we choose one of them as the *active segment* and we approximate it by its salient points. We then couple these salient points to the other segment (the *passive segment*) by linear interpolation. This initial match usually does not meet all the criteria stated in Section 4.1. For each criterion, the associated deviation is measured. This will be manifested in the form of a force acting on the coupling result. These forces may reinforce each other if they are in the same direction. They may also balance off each other if they are in opposite directions. During each iteration, we compute the net resultant force at each point and make corresponding adjustment to the coupling result. The adjusted result is then evaluated again using the same logic in an iterative manner.

Therefore, the force-driven approach resembles the steepest descent algorithm in that the direction of search follows the most promising direction towards the solution. Hence, correspondence matching under this force-driven framework is an iterative process to find the points of equilibrium where these forces canceled out each other.

4.3. *Force One: Deflection Force*

The sleepers criterion requires the set of sleepers to be approximately parallel or concurrent to each other. Based on the sleepers at the endpoints, for each salient point $C_i(S_p)$ we can estimate the expected orientation of the vector linking $C_i(S_p)$ to $C_j(f(S_p))$. We define a deflection force which serves to bring the orientation of coupled results to the expected orientation. In other words, it helps in minimizing the deviation in coupling orientation.

Referring to Fig. 1, the four points P_1 through P_4 are endpoints of the two simple segments shown. In an iteration, a salient point $C_i(S_p)$, denoted P_5, somewhere in between the two endpoints P_1 and P_3 is coupled to $C_j(f(S_p))$, denoted P_6. The point P_0 is defined as the intersection of

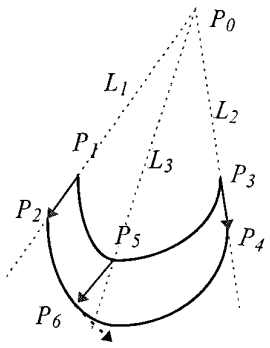

FIG. 1. Deflection force. Deviation between the coupled result and the expected orientation generates a deflection force at P_6 which is shown as a dotted arrow.

the two sleepers at the two ends of the simple segments. From P_0, we estimate the expected orientation of the sleeper for $C_i(S_p)$. The geometric relationships among them are

$$\begin{aligned} L_1 &= P_1 + t_1(P_2 - P_1) \\ L_2 &= P_3 + t_2(P_4 - P_3) \\ L_3 &= P_5 + t_3(P_5 - P_0). \end{aligned} \tag{6}$$

In our 2D case, the value t_1 corresponding to the intersection between L_1 and L_2 can be solved by

$$t_1 = \frac{\begin{array}{l}(P_3(x) - P_1(x))(P_4(y) - P_3(y)) \\ -(P_3(y) - P_1(y))(P_4(x) - P_3(x))\end{array}}{\begin{array}{l}(P_2(x) - P_1(x))(P_4(y) - P_3(y)) \\ -(P_2(y) - P_1(y))(P_4(x) - P_3(x))\end{array}}; \tag{7}$$

then, P_0 can be obtained as

$$P_0 = P_1 + t_1(P_2 - P_1). \tag{8}$$

The expected orientation of L_3 is described by $(P_5 - P_0)$. Hence, this is what we want to have,[1]

$$\langle (P_5 - P_0) \rangle \times \langle (P_6 - P_5) \rangle = 0. \tag{9}$$

Notice that t_1 cannot be solved when the denominator in (7) is approaching zero, which implies L_1 and L_2 are almost parallel; then we expect the orientation of L_3 to be identical to that for L_1, which is $(P_2 - P_1)$. Summarizing, we have the first criterion function defined as

$$q_1(S_p) = \begin{cases} \langle (P_5 - P_0) \rangle \times \langle (P_6 - P_5) \rangle & \text{if } \langle (P_2 - P_1) \rangle \cdot \langle (P_4 - P_3) \rangle \neq 1 \\ \langle (P_2 - P_1) \rangle \times \langle (P_6 - P_5) \rangle & \text{otherwise.} \end{cases} \tag{10}$$

The first unit vectors in the cross products are the expected orientations while the second unit vectors are the observed orientations. Therefore, for nonzero $q_1(S_p)$, if it is positive we should rotate the sleeper in a clockwise manner and vice versa. It turns out that the force direction can readily be obtained from the left–right attribute of the DDT's. Going back to Fig. 1, let us say that the passive curve is coded from P_2 to P_4; then the DDT at P_6 is a right tangent pointing in the direction toward P_4. Rotating a sleeper in the clockwise manner would mean to go in the direction opposite to that indicated by the DDT. However, if the passive curve is coded in the reverse manner, the DDT at P_6 is a left tangent pointing in the direction toward P_2. Consequently rotating a sleeper in the clockwise manner would mean to go in the direction indicated by the DDT and vice versa. Since applying translation to the two curves do not affect the signs of $q_1(s)$[2] and the left–right attributes of DDT's are invariant to rotation, the discussion given above holds in general. To summarize, the direction of the force exerting on $C_j(f(S_p))$ for nonzero $q_1(S_p))$ is given by

$$-g_1(S_p) = \begin{cases} \psi_j(f(S_p)) & \text{if } [q_1(S_p) > 0].\text{eq. IsLeft}(\psi_j(f(S_p))) \\ \psi_j(f(S_p)) + \pi & \text{otherwise,} \end{cases} \tag{11}$$

where the function IsLeft identifies if the given DDT is a left-tangent and the notation ".eq." is the equality operator. Notice that the expressions on both sides of the equality operators are Boolean expressions.

4.4. *Force Two: Tangent Alignment Force*

The tangent alignment force enforces the tangent alignment criterion discussed in Section 4.1. This force adjusts misaligned DDT's, to ensure similar DDT's at the two points being coupled. This force serves to minimize deviation between DDT's at the conjugate pairs. Note that we use different ranges to represent left- and right-tangents, both of these DDT's must be converted into left-tangents

[1] We are using only the z-component from the cross product as a measure of the vertical distance between the first unit vector and the endpoint of the second unit vector.

[2] The signs of the cross products will change if and only if the roles of the two curves are interchanged. That is, the active segment becomes the passive segment and vice versa.

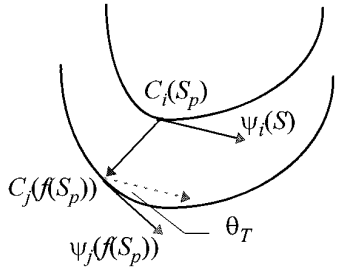

FIG. 2. Tangent alignment force. The angle θ_T defines the difference between the DDT's of the coupled points. This will generate the tangent alignment force at $C_j(f(S_p))$.

(or both right-tangents) before measuring the tangent deviation angle θ_T.

Referring to Fig. 2, during an iteration, suppose a salient point $C_i(S_p)$ is coupled to $C_j(f(S_p))$. The DDT's at these points are $\psi_i(S_p)$ and $\psi_j(f(S_p))$, respectively. Since their orientations are different, a tangent alignment force exists at $C_j(f(S_p))$ to bring the coupling towards the point where θ_T fades out to zero. The magnitude of this tangent alignment force is again proportional to the angle of deviation θ_T which can be obtained from the cross product

$$q_2(S_p) = \langle Left(\psi_i(S_p))\rangle \times \langle Left(\psi_j(f(S_p)))\rangle, \qquad (12)$$

where $\langle\theta\rangle$ denotes a unit vector with orientation θ. To determine the force direction, let us consider the example given in Fig. 3. Note that since all tangent vectors are cast to left-tangents, we can ignore the effect of coding direction for the time being. It is easy to verify that $q_2(s)$ will be positive if point s is coupled to a and negative when coupled to b. The correct tangent could be found somewhere in between a and b. Next, we can make use of the concavity information incorporated in DDT to identify the direction to move for a better match. If the DDT's along the passive segment are actually left-tangents, then its coding direction agrees with the orientation of the second unit vector in (12). Consequently, for positive $q_2(S_p)$ we should move forward along the passive segment and vice versa. However, if the DDT's along the passive segment are actually right-tangents, then its coding direction is opposite to the orientation of the second unit vector in (12). The conclusion is, for positive $q_2(S_p)$ we should move backward along the passive segment and vice versa. Based on the same argument as we did for the previous criterion, it follows immediately that for nonzero $q_2(S_p)$, the force direction at $C_j(f(S_p))$ is given by

$$-g_2(S_p) = \begin{cases} \psi_j(f(S_p)) & \text{if } [q_2(S_p) > 0].\text{eq. IsLeft}(\psi_j(f(S_p))) \\ \psi_j(f(S_p)) + \pi & \text{otherwise,} \end{cases} \qquad (13)$$

4.5. *Force Three: Approximation Error Force*

The similarity criterion aims to yield a good approximation of curve C_j by the set of coupled points $C_j(f(S_p))$. The accuracy of approximation is indicated by the largest perpendicular distance of the represented segment from the chord. The smaller this distance measures, the better is the approximation. This force is employed to minimize this error during coupling.

Referring to Fig. 4, during an iteration, two successive salient points $C_i(S_p)$ and $C_i(S_{p+1})$ are coupled to $C_j(f(S_p))$ and $C_j(f(S_{p+1}))$, respectively. Denote $d_j(f(S_p))$ the perpendicular distance measured on the farthest point on C_j from the chord curve defined by $C_j(f(S_p))$ and $C_j(f(S_{p+1}))$. The smallest possible approximation error is obtained when $d_j(f(S_p)) = 0$. For nonzero $d_j(f(S_p))$, we should be able to get a better approximation by moving the coupled points closer. Note that neighborhood information is required for evaluating this criterion.

In the force-driven model, whenever neighborhood information is required, we need to consider all immediate neighbors. In this case, for the salient point $C_i(S_p)$ we need to consider both neighbors $C_i(S_{p-1})$ and $C_i(S_{p+1})$. But since

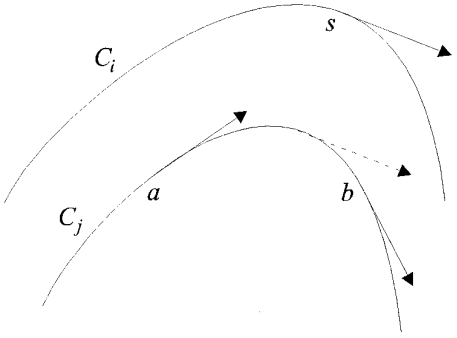

FIG. 3. Tangent alignment. The value of $q_2(s)$ will be positive if point s is coupled to a and negative when coupled to b. The correct tangent (dotted arrow) could be found somewhere in between a and b.

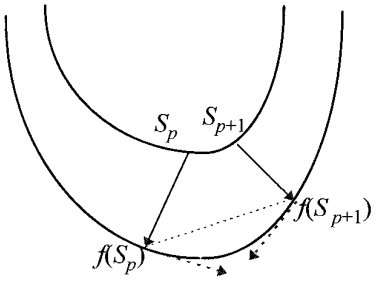

FIG. 4. Approximation error force. Large approximation error will generate approximation error forces to pull the coupled points closer. These forces are shown as dotted arrows here.

the coupling result at the latter point has been used to obtain $d_j(f(S_p))$, we need only to consider $C_i(S_{p-1})$. Now if $d_j(f(S_{p-1})) > 0$, there is a force at $C_j(f(S_p))$ to push it toward $C_j(f(S_{p-1}))$. Definitely, there is also a force at $C_j(f(S_{p-1}))$ to push it toward $C_j(f(S_p))$. In conclusion, we have at most two forces acting on $C_j(f(S_p))$, both tangentially but at opposite directions. Since we are interested in the net remaining force, we define the third criterion function as

$$q_3(S_p) = d_j(f(S_p)) - d_j(f(S_{p-1})). \tag{14}$$

The force direction depends on the sign of $q_3(S_p)$. For nonzero function values, if $q_3(S_p) > 0$, $f(S_p)$ should move closer to $f(S_{p+1})$ to reduce the value of the first term; otherwise it should move closer to $f(S_{p-1})$. Note that only segments with similar concavity should be coupled, it follows that the left–right attributes at the coupled segments are equal if they are coded in the similar manner. For the case given in Fig. 4, from the indices of the active segment, we can be sure that the top curve is coded from left to right. Now if the lower curve is also coded from left to right, then the left–right attributes of the DDT's at both segments are equal, both are right tangents. Moving towards $f(S_{p+1})$ would mean to follow the direction of DDT along the passive segment. However, if the lower curve is coded in the reversed manner, then the left–right attributes of the two segments are different, the DDT's at the both curve are left-tangents. Moving toward $f(S_{p+1})$ would mean to trace back to some earlier points (reverse the direction of DDT) along the passive segment. Generalizing the arguments, the force direction at $C_j(f(S_p))$ for nonzero $q_3(S_p)$ is given by

$$-g_3(S_p) = \begin{cases} \psi_j(f(S_p)) & \text{if } [q_3(S_p) > 0].\text{eq. } [\text{IsLeft}(\psi_i(S_p)).\text{eq. IsLeft}(\psi_j(f(S_p)))] \\ \psi_j(f(S_p)) + \pi & \text{otherwise.} \end{cases} \tag{15}$$

4.6. *Force Four: Elasticity Force*

The uniform stretching or contraction requirement described in the elasticity criterion is modeled by means of an elasticity force. The force attempts to minimize nonuniform lengthening or shortening of curve segments represented by successive salient points on the active segment when projected onto the passive segment through coupling. To remove the need for computing global information in order to access uniformity, we take advantage of the force-driven approach which allows forces to counteract on each other and consider only two successive salient points when defining the elasticity force.

Referring to Fig. 5, during an iteration, two successive salient points $C_i(S_p)$ and $C_i(S_{p+1})$ are coupled to $C_j(f(S_p))$ and $C_j(f(S_{p+1}))$, respectively. In general,

$$\text{arcLength}_i(S_p, S_{p+1}) \neq \text{arcLength}_j(f(S_p), f(S_{p+1})).$$

If we approximate the segments by a deformable nonrigid elastic string, this change in segment length requires applying to it an external force (either stretching or compression). The string will react with a force opposite to the external force. The relation is modeled using Young's law of elasticity, which is

$$T = \frac{\Delta L}{L_0} \cdot A \cdot Y, \tag{16}$$

where T denotes the tensile strength or stress experienced, L_0 denotes the original length (which is the arc length between $C_i(S_p)$ and $C_i(S_{p+1})$, ΔL is the change in length (the difference between the two arc length measures, A is the cross section of the object, Y is the Young's modulus for the material. Assuming uniform cross sectional area, the equation reduces to

$$T = \frac{\Delta L}{L_0} \cdot k, \tag{17}$$

where k is an arbitrary constant and both ΔL and L_0 are in pixels. Share this force equally between $C_j(f(S_p))$ and $C_j(f(S_{p+1}))$, the magnitude of reaction occurred at each point is

$$\frac{\Delta L}{L_0} \cdot \frac{k}{2}. \tag{18}$$

Following the same derivation for the similarity criterion, we need to take in another force from the error measured with the immediate neighbor on the other direction. Besides, since we shall assign a weight to the criterion function

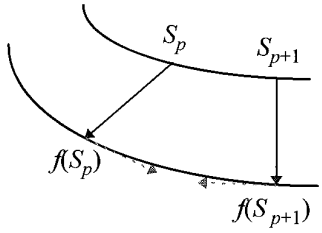

FIG. 5. Elasticity force. Stretching (or compression) of active segment when coupled to passive segment creates elasticity forces to bring the coupled results closer to (or further apart from) each other.

anyway, we can drop the constant $k/2$ to arrive at the third criterion function,

$$q_4(S_p) = \frac{\text{arcLength}_j(f(S_p), f(S_{p+1}))}{\text{arcLength}_i(S_p, S_{p+1})} - \frac{\text{arcLength}_j(f(S_{p-1}), f(S_p))}{\text{arcLength}_i(S_{p-1}, S_p)}. \quad (19)$$

Again, the force direction depends on the sign of $q_4(S_p)$. For nonzero function values, if $q_4(S_p) > 0$, $f(S_p)$ should move closer to $f(S_{p+1})$ so as to reduce the first term while at the same time raise the second term; otherwise it should move closer to $f(S_{p-1})$. Following the same argument as for the previous force formulation, the force direction at $C_j(f(S_p))$ for nonzero $q_4(S_p)$ is given by

$$-g_4(S_p) = \begin{cases} \psi_j(f(S_p)) & \text{if } [q_4(S_p) > 0].\text{eq. } [\text{IsLeft}(\psi_i(S_p)).\text{eq. IsLeft}(\psi_j(f(S_p)))] \\ \psi_j(f(S_p)) + \pi & \text{otherwise.} \end{cases} \quad (20)$$

4.7. Force-Driven Coupling at Work

At each iteration, for each salient point $C_i(S_p)$ in curve C_i, we have a point in C_j coupled to it. We denote the matching schemes at the nth iteration by $f(S_{p,n})$. For each point $C_j(f(S_{p,n}))$, we calculate the four forces at $C_j(f(S_{p,n}))$ and combine them to give a resultant force $\tau_{p,n}$ at the position. Determination of these force directions is straightforward as can be seen from the formulations. Combining these forces is also very simple because they all act tangentially at $C_j(f(S_{p,n}))$, hence requiring only simple addition or subtraction on the magnitudes. The direction of the resultant force $\tau_{p,n}$ is still tangential at $C_j(f(S_{p,n}))$.

The direction of the resultant force tells us the direction to move in order to search for a better coupling result. However, knowing the direction to go does not tell us the location of a better coupling point. To determine the amount of shift, we can apply line search techniques because they are simple and numerically stable. Here we have at least three choices. One is to adopt the greedy approach, where we look for the coupling point at which the resultant force will be minimized. Observe that the greedy search is localized and parallel; during an iteration, each salient point looks for its best coupling point based on its own current coupling result and its immediate neighbors'. The second method is to find a point in the shifting direction with a resultant force of smaller magnitude. Basically we can apply the same greedy method but terminate the search once a coupling point with smaller resultant force is found. The third method is to derive the amount of shift from the magnitude of the resultant force.

To search for the greedy solution, either the golden section search or Peters and Wilkinson's method [19] (which combines both linear interpolation and golden section search) can be used. In our experiment, we tried all three shifting strategies and we shall discuss their pros and cons in Section 5.4. But no matter which shifting strategy is used, the global monotonous coupling constraint restricts the range of search. We must limit the range of search such that adjacent sleepers do not cross over each other. One obvious choice is to limit the range of search to one-third of the distance to the nearest salient point in the shifting direction. Hence, we define

$$\text{arcLengthToNext } SP_{p,n} = \begin{cases} \text{arcLength}(f(S_{p,n}), f(S_{p+1,n})) & \text{if } \tau_{p,n} = \psi_j(f(S_{p,n})) \\ \text{arcLength}(f(S_{p-1,n}), f(S_{p,n})) & \text{if } \tau_{p,n} + \pi = \psi_j(f(S_{p,n})). \end{cases} \quad (21)$$

In our implementation, when the greedy approach is involved, we limit the search window to that range. When the third approach is used, we used the following simple rule to determine the shift at point $C_j(f(S_{p,n}))$:

$$\text{shift}_{p,n} = \min\left(\tfrac{1}{3}\,\text{arcLengthToNext } SP_{p,n}, \text{magnitude}(\tau_{p,n})\right). \quad (22)$$

The coupling results are updated using these rules:

$$\begin{aligned} f(S_{p,n+1}) &= f(S_{p,n}) && \text{if } C_i(S_p) \text{ is a tsegment boundary} \\ f(S_{p,n+1}) &= f(S_{p,n}) + \text{shift}_{p,n} && \text{if } \tau_{p,n} = \psi_j(f(S_{p,n})) \\ f(S_{p,n+1}) &= f(S_{p,n}) - \text{shift}_{p,n} && \text{if } \tau_{p,n} + \pi = \psi_j(f(S_{p,n})). \end{aligned} \quad (23)$$

The first rule prohibits coupled segment endpoints from shifting at all. The remaining two rules adjust new coupling results in the direction as indicated by $\tau_{p,n}$. Obviously, the process terminates when for all p, $C_j(f(S_{p,n+1})) = C_j(f(S_{p,n}))$. That is, when the coupling results stabilize.

5. THE DISCRETE ALGORITHM FOR DETECTING PERCEPTUAL PARALLELISM

This section summarizes our discrete algorithm for parallel curve detection. The algorithm is capable of detecting perceptual parallelism and partial parallelism and is outlined below:

Step 1. *Decomposition of curves into simple segments.*

Step 2. *Prescanning for possible coupling between simple segments.* Unlike the continuous case, we allow two simple segments (one from each curve) to be coupled if their DDT's are equivalent within a given threshold (see Section 5.1) at both endpoints.

Step 3. *Force-driven coupling.* Find salient points for C_i (the active curve) and fit these salient points in each segment in C_i to the corresponding segment in C_j (the passive curve) by interpolation. Apply force-driven coupling to the fitting process until the coupling results stabilize (see Section 4).

Step 4. *Parallelism verification.* Identify the portions that are parallel (see Section 5.3.)

Step 5. *Parallelism reporting.* Keep the combination that gives the longest continuous parallel section. Report parallelism if the parallel section is longer than a certain threshold (in our case, 40% of curve length).

Next we present some of the practical issues associated with the algorithm.

5.1. *Fuzziness in Prescanning Step*

The intention to detect parallelism under perspective projection and perceptual parallelism requires us to relax the coupling requirement in Step 2. We must add some fuzziness to the prescanning mechanism by giving some deviation allowance between the underlying DDT's. In other words, we modified the DDT equivalency definition given previously in (4) to

$$\langle \text{Left}(\psi_1)\rangle \cdot \langle \text{Left}(\psi_2)\rangle \geq \cos(\varepsilon_{DDT}) \Leftrightarrow \psi_1 \triangleq \psi_2. \quad (24)$$

The threshold ε_{DDT} we set would affect the amount of disruption we allow. In our experiment, we set this ε_{DDT} value to $\pi/6$ (30°).

5.2. *Locating Salient Points*

In the literature, there exists a large amount of research work carried out under the topics ''detection of significant points'' [20] or ''polygonal approximation'' [2]. Among them, the simplest approach is the method of *iterative endpoint fit and split* described by Duda and Hart [5]. As we mentioned earlier, we do not intend our algorithm to depend heavily on the accuracy of the underlying salient point detection algorithm; efficiency is of primary concern. Therefore, we adopt Duda's approach with a slight enhancement to include critical points (zero-curvature points and curvature extrema) in the set of salient points. It is worth mentioning here that based on the rationale described in Section 3.1, our algorithm is quite insensitive to the stability of the underlying salient point detection algorithms. We simply want to achieve reduced computation by using salient points in the correspondence coupling phase.

To detect critical points, we need tangent and curvature estimates. Based on the same rationale, we studied several tangent and curvature estimation algorithms and introduced a conditioning algorithm [31] for the detection of zero-curvatures. We first apply the scheme to detect zero-curvature points and segment the curve at these zero-curvature positions. Between two successive zero-curvature points, we pick the point with the maximum absolute curvature magnitude as the curvature extremum for that segment. At the maximum curvature point, we further break the segment in two. For a curve with no zero-curvatures, we simply break the curve at the point where it has the peak curvature magnitude.

For each segment, we apply the iterative endpoint fit and split algorithm to detect all salient points. It is obvious that most of these simple segments have only a small number of salient points, except for the segments containing undetected zero-curvature points. This is an important property for the iterative force-driven coupling process. With less salient points to work with, doubtless the process requires less computation time.

5.3. *Checking for Parallelism in the Coupling Result*

As hinted by discussions in Section 3.2, we need a new strategy to report parallelism between each pair of segments. After the force-driven coupling phase, we expect the following if C_i and C_j are perceptually parallel:

- for all p, the orientation from $C_i(S_p)$ to $C_i(S_{p+1})$ should be similar to the orientation from $C_j(f(S_p))$ to $C_j(f(S_{p+1}))$
- curve C_j can be approximated by lines joining successive $C_j(f(S_p))$ with the same degree of accuracy as curve C_i is approximated by successive $C_i(S_p)$.

Therefore, to detect parallel segments after coupling, we need first to make sure that the orientations described in the first property above fall within an allowable limit. If this property is satisfied, then we proceed to measure how big the error is when we approximate the segment between $C_j(f(S_p))$ and $C_j(f(S_{p+1}))$ by the straight line joining them. We apply the following two tests to every segment delimited by successive salient points. If it satisfies either one of the tests, we treat the approximation as acceptable. The two tests are:

- check for good proximity: for all points in between $C_j(f(S_p))$ and $C_j(f(S_{p+1}))$, the maximum distance from chord allowed is

$$\text{errorThreshold} \times \frac{\text{chordLength}_j(f(S_p), f(S_{p+1}))}{\text{chordLength}_i(S_p, S_{p+1})} \quad (25)$$

and we must make sure the condition is satisfied by com-

paring the value of $d_j(f(S_p))$ against this derived limit. The factor *errorThreshold* in (25) is the same parameter used in the *iterative endpoint fit and split process* mentioned in Section 5.2.

• check for good continuity in DDT's: if we think of using the unit vectors $\langle \text{Left}(\psi_j(f(S_p)))\rangle$ and $\langle \text{Left}(\psi_j(f(S_{p+1})))\rangle$ to divide a circle into two regions, then the left-equivalent DDT's for all points in between must fall into the same region for good continuity to occur. The simplest way to carry out the test is to treat orientations as unit vectors and apply the positively oriented-vector test given in (2).

The first test is applied before the second one because the former test is faster to compute. Only if the first test fails then the second test is executed.

5.4. *Convergence and Complexity Issues in Force-Driven Coupling*

The force-driven approach is an iterative technique designed to model the interactions among several criteria. The process terminates when all points are at a steady state. Since the mechanism resembles the steepest descent method, convergence is guaranteed if there exists such a solution and the shift between each iteration is sufficiently small so that we do not overshoot. In practice, we would like to make the biggest possible shift in each iteration so that we can arrive at the solution with a smaller number of iterations. This can be achieved using a greedy search, an early-terminate-greedy search, or heuristic mapping based on force magnitude as noted in Section 4.7. The choice of the searching mechanism affects both convergence and complexity. The complexity for the heuristic mapping is the most promising as it is of $O(n)$, where n is the number of salient points in the active curve. The complexity for a greedy approach using the golden section search is of $O(n\log_r\rho)$, where r is 1/golden-ratio and is approximately 1.618, and ρ is

$$\tfrac{1}{3}\ \text{arcLengthToNext}\ SP_{p,n}$$

as defined by (21). The worst-case complexity for early-terminate-greedy is the same as that for greedy; this occurs when the current coupled point is the best choice within the given range. Nevertheless, no matter which searching technique is used, the complexity is *better* than that which can be achieved using greedy search in an active contour formulation, which is of complexity $O(mn)$ as described in Section 2. The fact lies in that in the force-driven mechanism, determination of the gradient directions for each criterion or constraint is done separately.

Concerning convergence, conceptually we feel more comfortable with the first two searching techniques because the expected force with which the next iteration starts is smaller than the current one. However, in practice, when only local information is used and all coupling results are updated in parallel, there is no guarantee that oscillation can be avoided. For the heuristic mapping method oscillation is unavoidable if the weights are given very large values such that a small derivation from equilibrium will generate strong forces. Fine tuning on weights is required to make sure very big jumps do not occur. Large weight values will have minor impact, if any, for the greedy search.

The reason for oscillation is that we skip too many points in one jump. Furthermore, oscillation suggests that the solution should be somewhere in between. A simple application of this observation is to force the coupling for the next iteration to be the midpoint of the oscillation range. To detect oscillation we need to keep three values: the previous, the current, and the move-to coupling locations. If they are monotonically increasing or decreasing, oscillation does not occur and we are moving in the right direction. However, if the current location is a maximum or a minimum, we are going back and forth; then the coupling location to be tested in the next iteration will be the average of the current and the move-to values. With this simple mechanism for adjusting the coupling locations at the end of each iteration, oscillation will not proceed forever. Formally, this can be described by

$$\text{if either}\ \begin{cases} f(S_{p,n-1}) > f(S_{p,n}) < f(S_{p,n+1}) \\ f(S_{p,n-1}) < f(S_{p,n}) > f(S_{p,n+1}) \end{cases},$$

$$\text{then}\ f(S_{p,n+1}) = \frac{f(S_{p,n}) + f(S_{p,n+1})}{2}.$$

With oscillation prevented, it is still advisable to limit the number of iterations allowed because with nonparallel sections, we need more iterations to arrive at the equilibrium. From our experiments, we deduced that if parallelism does exist, we rarely need more than 10 iterations for a correct detection. But if parallelism does not exist, we may need more than 20 iterations before the result converges to a stable state. As a conclusion, it is sensible to limit the number of iterations to a value around 10.

For general purpose applications, we suggest the early-terminate-greedy approach because it allows faster decisions, although it may take more iterations to converge when compared with the greedy approach. For special applications, if one can fine tune the weights such that for most of the time, the step size at each shift is quite small (say, within 5% of image diagonal length), the third approach is more efficient because we do not need to evaluate the resultant forces at several candidate points before making each decision.

5.5. *Choice of Weights in Force-Driven Coupling*

In the force-driven framework, the weights assigned to the constraints will affect the final solution for the mapping

function f. First, we should assign heavier weights to hard constraints to make sure that they are properly entertained in the final result. A better alternative is to enforce the hard constraints by using them to limit the search space. Second, the relative importance amongst these constraints are reflected in the weights given to the associated functions. A third consideration is that the constraints may be formulated in different units and the assignment of these weights must somehow normalize this effect.

Therefore, the process of obtaining these weights is quite straightforward. Observe that it is quite common to have the constraint functions defined in different units. Hence, the very first thing to do is to normalize the effects due to this difference. This can be achieved by assigning some arbitrary initial weights to w_i and collect statistics on the forces produced. In this phase, only the magnitudes of these forces are important as our objective is to obtain the ratios amongst these forces such that we can scale them to have magnitudes comparable with each other. The policy here is to prevent any particular constraint to dominate the optimization process. This process of normalizing the relative constraint parameters has been adopted in other optimization techniques that attempt to look for the optimal solution in a multiple parameter space. Typical examples are regularization techniques and active contour models [11].

Once these forces are scaled to similar order of magnitudes, the next step is to fine tune these weights *if the relative importance amongst the constraints is known.* Very often this information on relative importance is not available. Most researchers would skip this fine-tune process and assign equal priority to these constraints. Others would adopt well-studied principles like the minimax strategy [11]. Alternatively, we can tune these weights experimentally or use training samples as in a neural network to recover the relative priority among these constraints and criteria.

In our experiment, we start off the process by arbitrarily setting the initial weights $w_i = 100$ and feeding the algorithm numerous test cases to obtain the following ratio amongst the force magnitudes: 5 : 15 : 150 : 25. Hence, the weights should be in the ratio 150 : 50 : 5 : 30 to fulfill the normalization purpose. We put an additional requirement that $d_j(f(m))$ must be greater than 2 to produce a move; hence, we set the weights at $w_1 = 15$, $w_2 = 5$, $w_3 = 0.5$, and $w_4 = 3$.

In the context of perceptual parallelism detection, we lack biological data to fine tune these weights. We tried to adjust these weights separately, but either no noticeable differences could be obtained or else the results would be degraded. Therefore, we ignored the fine-tune process and left the weights as they were. Note that if the greedy search is applied to get the amount of shift between iterations, then only the ratios among these weights are important. But if the amount of shift is derived from the magnitude of the resultant force, then the actual figures are important as well.

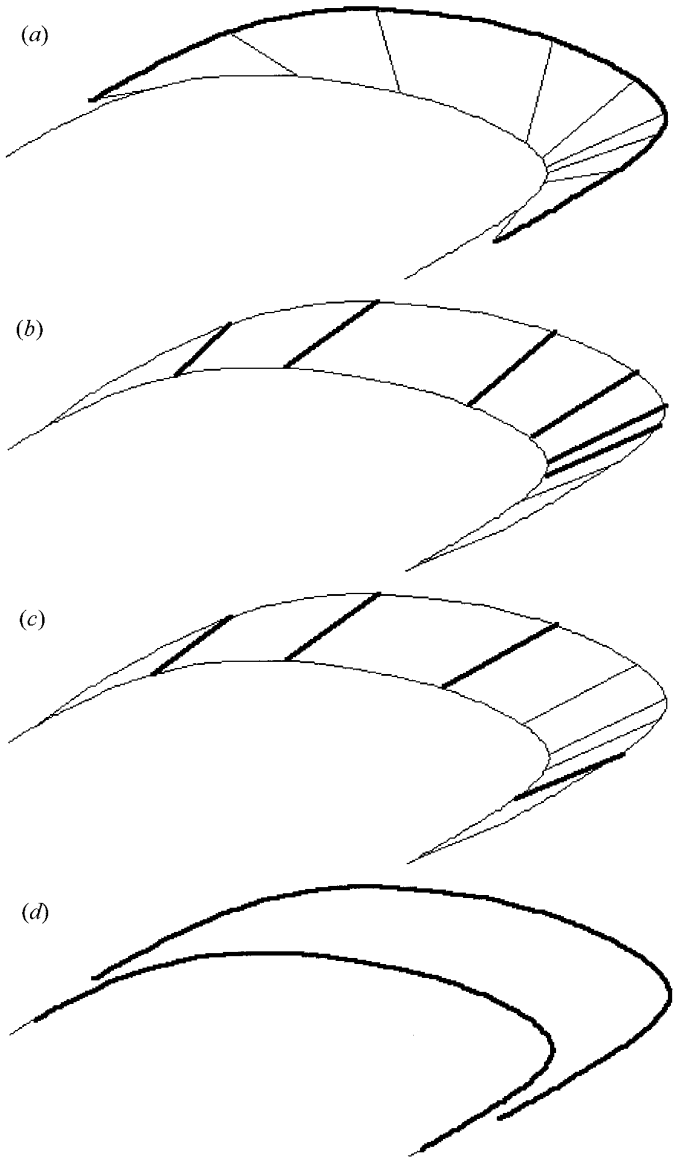

FIG. 6. An example of perspective projected parallelism: (*a*) shows the initial mapping with the active curve thickened; (*b*) shows the mapping after three iterations, thickened sleepers indicate changes in that iteration; (*c*) shows the correspondence established after seven iterations; (*d*) detection result with parallel sections thickened. Amount of shift in each iteration is determined from the greedy approach.

6. EXPERIMENTAL RESULTS

Several examples are shown in Fig. 6 to Fig. 13. All but the last three examples are synthesized data to illustrate the key features and robustness of the algorithm. In the last three examples, real image data are used. In all these

examples, we assigned $w_1 = 15$, $w_2 = 5$, $w_3 = 0.5$, and $w_4 = 3$. We showed these examples with different shifting strategies to illustrate that the result of detection is quite insensitive to the shifting strategy applied, although the number of iterations varied. To reduce the number of iterations required, instead of a simple interpolation as described in Step 3 of Section 5, we fit salient points in C_i to the nearest salient point in C_j with equivalent DDT's. Only for those salient points which failed to find their correspondence will be interpolated accordingly.

Figure 6 shows an example of a pair of parallel curves which have undergone perspective projection. The initial mapping are shown in (a) with the active curve thickened. Because initialization bases only on tangent information, the result is not really that good. Figure (b) shows the result after three iterations. The coupling result is more evenly distributed. Changes in the coupling pattern at the third iteration are shown in thick. After seven iterations the optimization process terminates with a promising correspondence scheme on human inspection. This is shown in (*c*) with changes in the coupling scheme at that iteration drawn in thick line. Parallelism was successfully detected and the result is shown in (d) with the parallel sections thickened. To illustrate that the algorithm is capable of withstanding instability in salient point selection schemes, we reduce *errorThreshold* to come up with a new set of salient points in Fig. 7. Although the number of iterations required may differ, parallelism is correctly identified.

Figure 8 shows an example of perceptually parallel curves. Figure (a) shows the initial matching with active curve and sleepers separating different simple curve segments shown in thick. Notice that the divide-and-conquer approach helps in ensuring the tangent orientation within each segment to be unique and prohibiting the possibility of multiple coupling candidates. In Fig. 9, another example of perceptually parallel curves is given. We show here an example with wrongly filtered zero-curvature coupling at the prescanning phase (step 2 of algorithm). The two zero-curvature points on the upper half of the curves are not coupled because the deviation of their DDT's is larger than the given threshold of 30°. Each curve is subdivided only into two at the thickened sleeper shown in figure (a). Observe the changes in the couplings from (a) to (e) which show how the force-driven coupling mechanism helps to provide a better coupling result for parallelism detection, in particular, the improvements at the two ends.

Figure 10 demonstrates the value of the algorithm to the detection of partial parallelism. Partial parallelism is much easier if the shorter curve is a subset of the longer curve. But we show in this example the more difficult case, to locate the common subset. In this example, the active curve is the smoothly varying curve while the passive curve is the curve with a hairpin turn. The salient points at the right-hand end of the active curve are ignored because

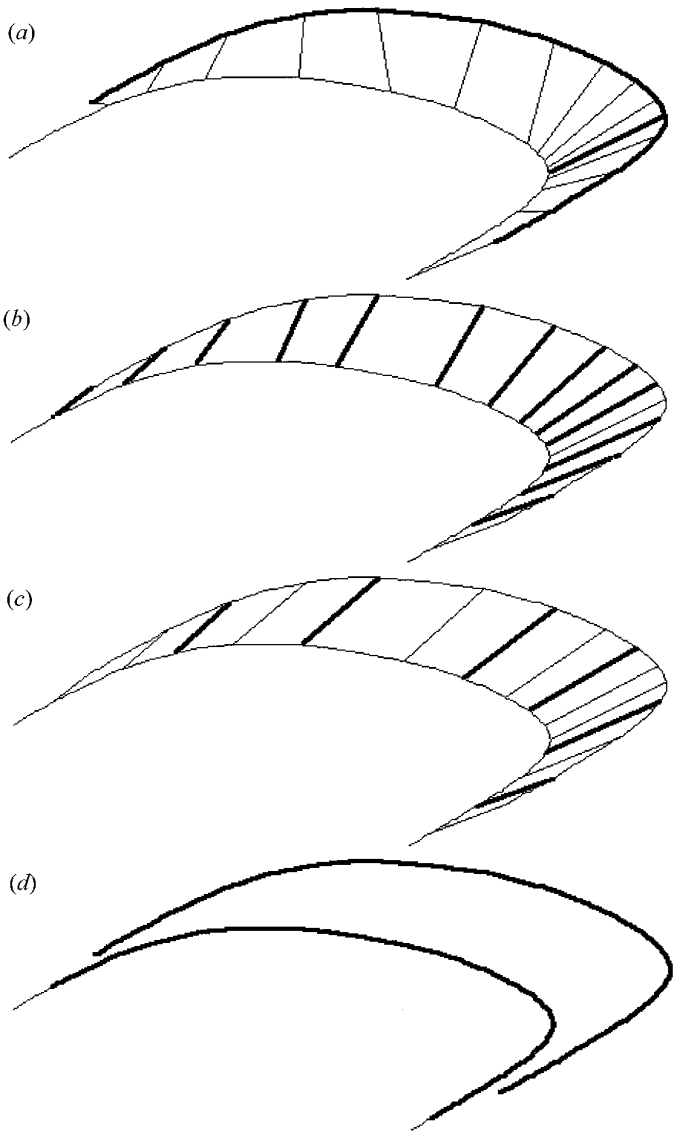

FIG. 7. The same set of curves but with different sets of salient points. This is included to illustrate the ability of the algorithm to tolerate instability in salient point detection. By reducing the distance threshold (*errorThreshold*), the curves are approximated by more salient points. Notations follow those for the previous figure: (c) shows the result after nine iterations; (d) shows the final coupling after seven iterations and the parallelism detected in thick.

they fail to find their corresponding salient points in the passive curve, and when they are interpolated, their sleepers are crossing each other. The common subset was roughly but correctly identified in the initialization process. Refinement was done in the force-driven optimization process and the parallel segments were successfully detected. In this example, the heuristic method (amount of shift derived from force magnitude) was adopted in the line search to determine the amount of shift in each iteration.

One point worth noting here is that the algorithm is

asymmetrical in the sense that we may get slightly different results when the roles of the two candidate curves are swapped. This asymmetry arises because we initialized the process and performed the optimization with respect to the shape of the active curve. To conform to the general notion about parallelism we may apply the algorithm twice (the first round treats curve A as C_i and the second round treats curve A as C_j) and combine the parallelism detected (an OR-operation) to give the final result. The results of applying the algorithm to real images were produced in this manner.

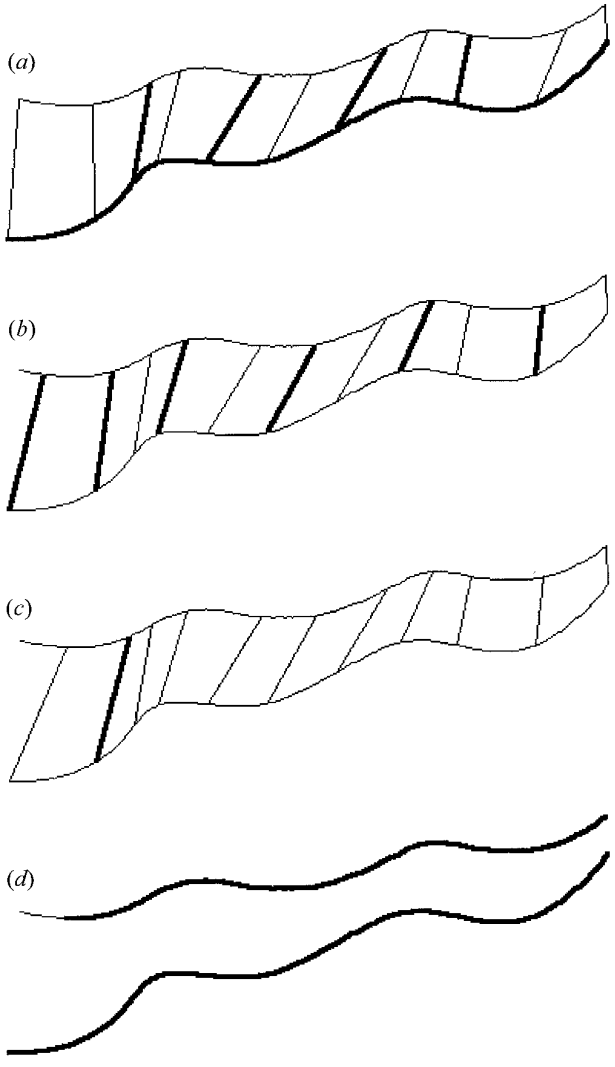

FIG. 8. An example of perceptual parallelism detection: (*a*) shows the initial matching with active curve and sleepers separating different simple curve segments shown in thick; (*b*) shows the results after the first iteration with changes in coupling results shown in thick; (*c*) shows the final coupling after three iterations and parallel sections detected are shown thickened in (*d*). Parallelism detection is successful for the whole span. Amount of shift in each iteration is determined using the greedy approach. Execution time on a pentium 90 is 0.8 s.

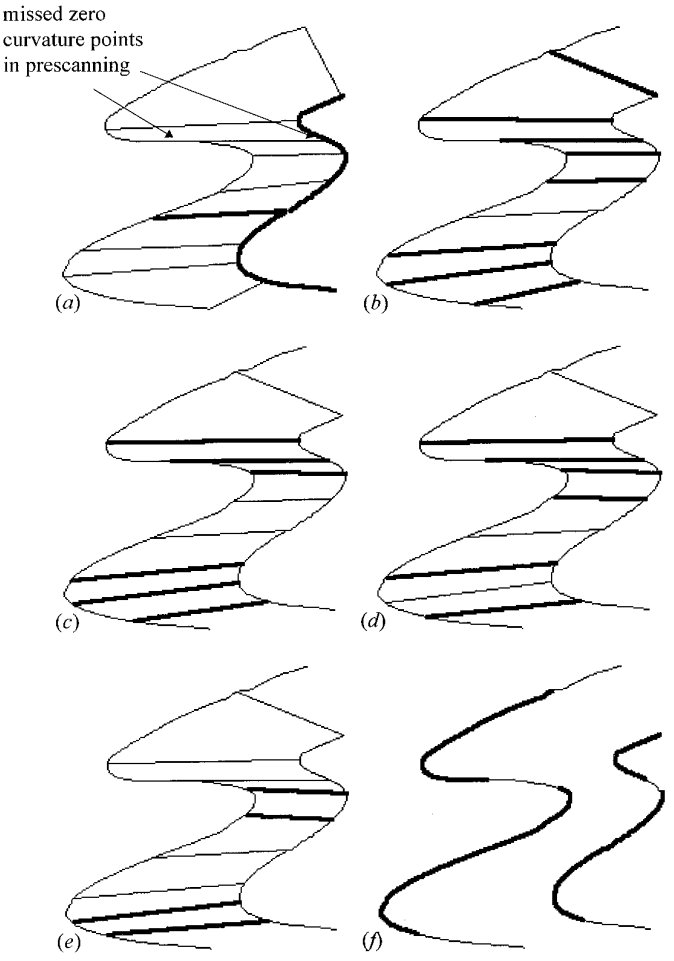

FIG. 9. An example with wrongly filtered zero-curvature coupling during prescan: (*a*) shows the initial matching with active curve and sleepers separating different simple curve segments shown in thick. Because the deviation at the tangents at the upper pair of zero-curvature points is larger than the allowed threshold, they are not coupled. As shown in this example, nonmatched zero-curvature points might have coupled had we allowed for a larger deviation threshold; (*b*) through (*e*) show the coupling scheme after each iteration; (*f*) shows the final detection result with parallel sections detected in thick. Amount of shift in each iteration is determined using the greedy approach. Detection is successful for a good range. In particular, observe the improvement of the coupling scheme at the two ends. Execution time on a pentium 90 is 0.8 s.

In each of these examples, the top image shows the input picture, the middle image shows the edge map and the edge labels, while the bottom image shows the results of the parallelism detection algorithm. In all these cases except the last, contours are extracted based on gradient measures. Contours in the last example are hand-traced. Contours with less than 30 pixels are removed. And for ease of viewing, the intensities at the bottom images are shifted to use only the upper half of the 256 gray levels, while the parallel curve segments detected by the algorithm are shown in black.

An example of applying the algorithm to an aerial photo is shown in Fig. 11. An example of applying the algorithm to a CT-scan is shown in Fig. 12. In Fig. 13, the algorithm is applied to a countryside scene. Details on the parallelism detection results are given in Table 1. Curves are represented in chain code and the number of codes for each curve C_i is shown in the third column. The best parallel curve C_j as viewed from a given curve C_i is shown in the fourth column. The mapping scheme describing parallel sections detected from the two curves is given in the last column, using the format "a..b → c..d". The meaning of this can be interpreted as points $C_i(x)$, $a \le x \le b$, are mapped monotonically to $C_j(y)$, $c \le y \le d$. From these results, a number of comments can be made. First of all, the algorithm is aimed for emulating the perceptual parallelism detection ability in the human vision system and is, therefore, more adapted to the detection of parallelism at more global scales. In general, noise has stronger effects on short segments. The effect of noise is particularly noticeable at the two ends of short open curves. The algorithm works very well for long curves. Second, we did not match parallel lines in the implementation, which is outside the scope of this study. Third, the success of the algorithm relies on the success of the contour tracking algorithm. At present, the algorithm works well for objects with good contrast and good isolation.

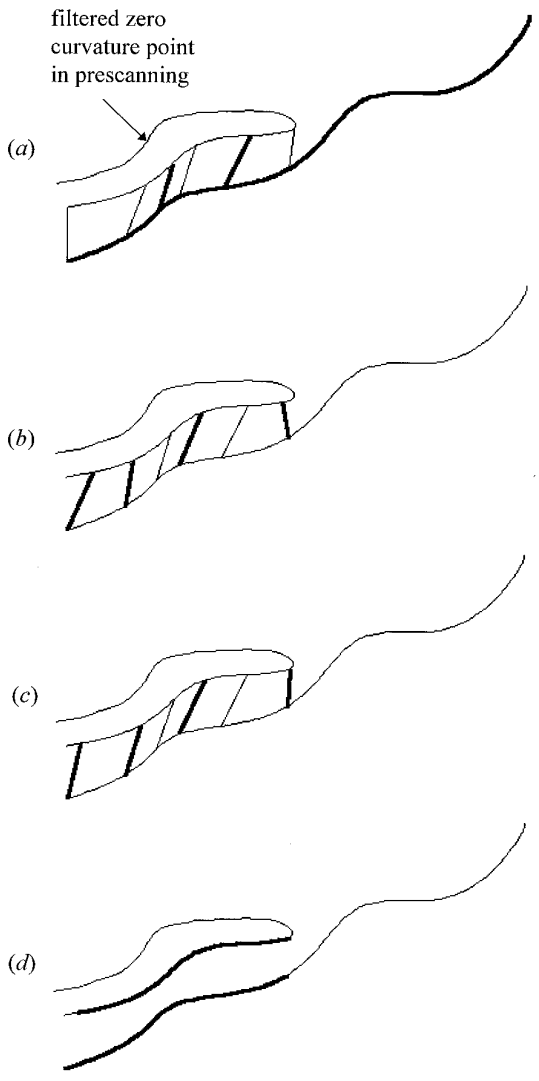

FIG. 10. An example of partial parallelism detection: (*a*) shows the initial matching with active curve and sleepers separating different simple curve segments shown in thick. As in the previous example, the deviation at the tangents at the upper pair of zero-curvature points is larger than the allowed threshold, they are not coupled; (*b*) shows the final coupling results after two iterations, changes in the coupling scheme at the iteration are shown in thick; (*c*) shows parallel section detected in thick. Parallelism detection is successful for the whole span. Amount of shift in each iteration is determined from the semi-greedy strategy. Execution time on a pentium 90 is 0.5 s.

7. DISCUSSION

Detecting perceptual parallelism is our ultimate objective; in addition, we aim to establish correspondence between the matching curves. We believe the solution to such problems can serve as a foundation to solve more difficult problems like shape matching, stereo correspondence, temporal correspondence for optical flow and motion tracking. In this work, we develop an algorithm which succeeded in finding a solution that would remain stable when given inaccurate tangent and curvature estimations and imperfect salient point detectors. We also introduced a new optimization technique by formulating a set of criteria as forces to guide the search for the solution that minimizes the total deviations measured by these criteria. We incorporated this force-driven process into the correspondence coupling step in our basic parallelism detection algorithm [33]. The strong point of this coupling mechanism is that it integrates easily several sources of information to obtain a solution which balances all perceptual criteria as discussed in Section 4.1. In this way, we take into account uncertainty in tangent and curvature measures and critical points detection implicitly.

In the force-driven coupling process, only local information is used. All forces can be calculated using information at each salient point, together with its immediate neighbors. This characteristic is important for an efficient implementation on parallel machines. In addition, all forces are along tangent directions and this implies the combination of force vectors to yield a net resultant force that involves only simple arithmetic. The overall efficiency is greatly enhanced by the divide-and-conquer strategy which splits curves into simple segments. Initial coupling at the segment endpoints, in fact, filtered out numerous trials that would have been done if we had adopted the voting-like approaches in the literature.

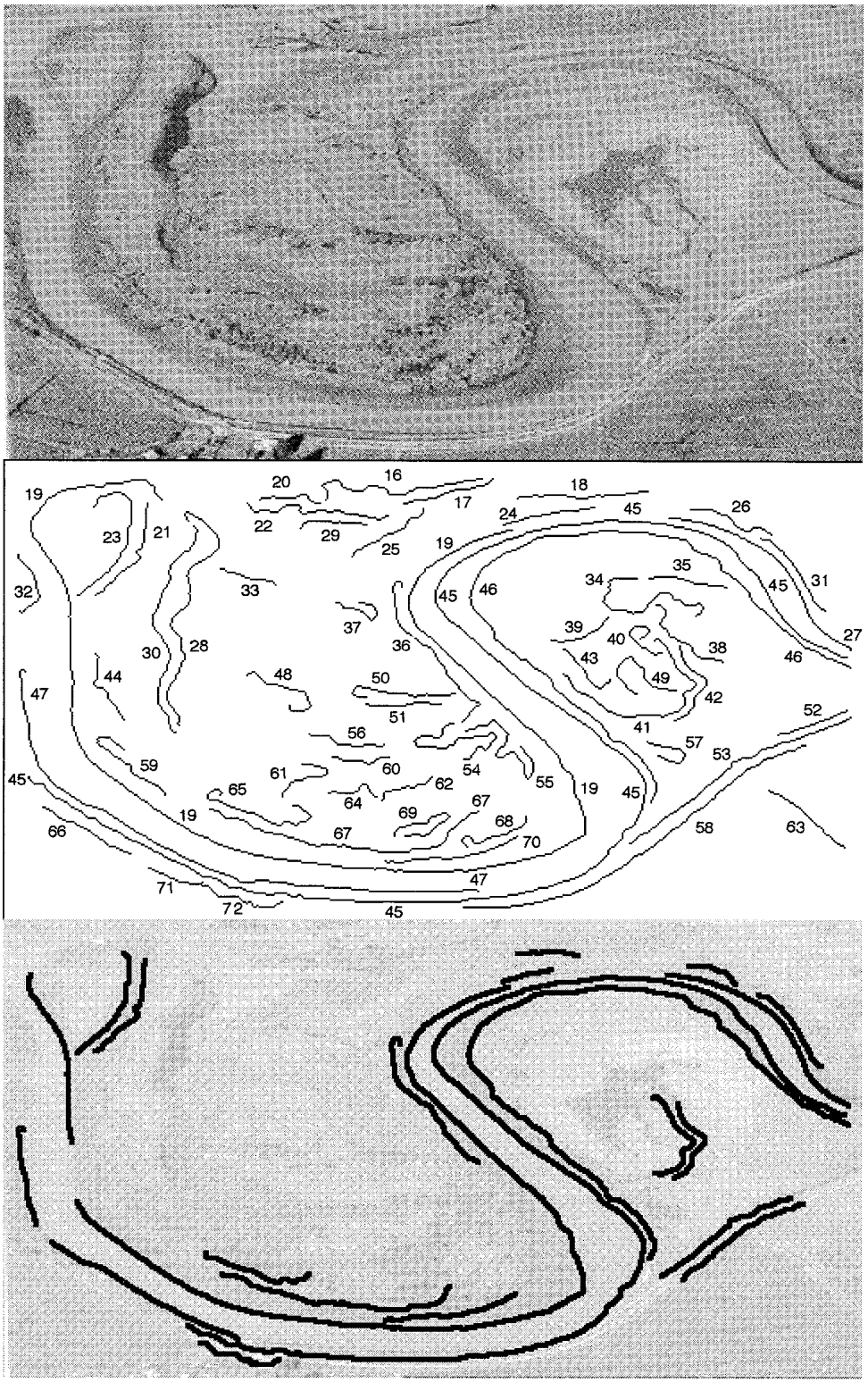

FIG. 11. An example of applying the algorithm to real image data: (top) shows the original image; (middle) shows the edge map; (bottom) shows parallel curves detected in thick black curves. Greedy algorithm is used in determining the amount of shift in each iteration. Maximum number of iterations is limited to 10.

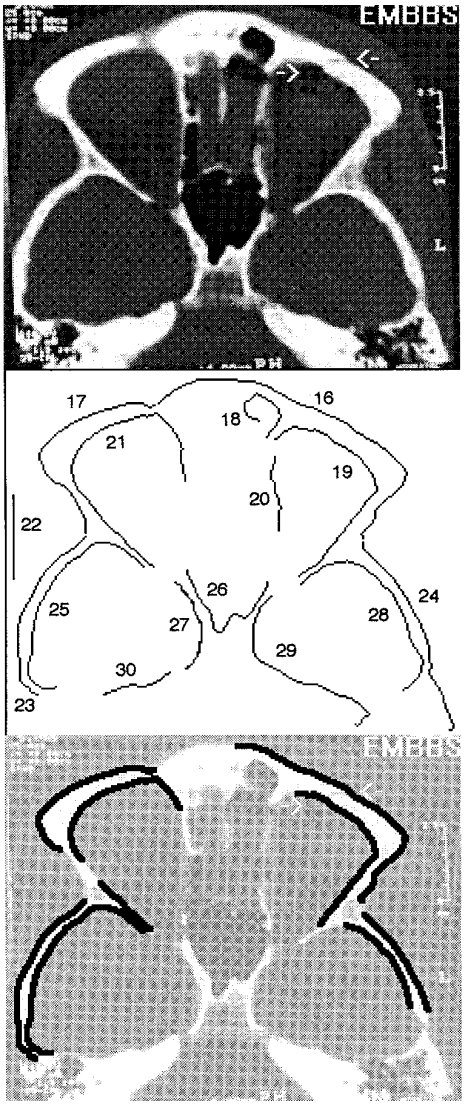

FIG. 12. An example of applying the algorithm to a medical image: (top) shows the original image; (middle) shows the edge map; (bottom) shows parallel curves detected in thick black curves. Greedy algorithm is used in determining the amount of shift in each iteration. Maximum number of iterations is limited to 10.

To further reduce computational needs, we applied the proximity constraint to parallelism detection. Here again, we used the idea of a perceptual window [4] and we varied the window size according to the length of the image diagonal. This idea is very effective in a multiresolution setting. Premature termination strategies like detecting good coupling before termination, checking for crossings of coupling results (not discussed in full because of the length) were also applied in the implementation.

The algorithm we put forth here can be easily modified to perform silhouette-based object recognition. The mechanism is invariant to translation and moderate scaling. To achieve rotational invariance, we need an efficient strategy to look for the most promising rotation parameters from the distribution of and the attributes associated with the feature points. This could be obtained by heuristic meth-

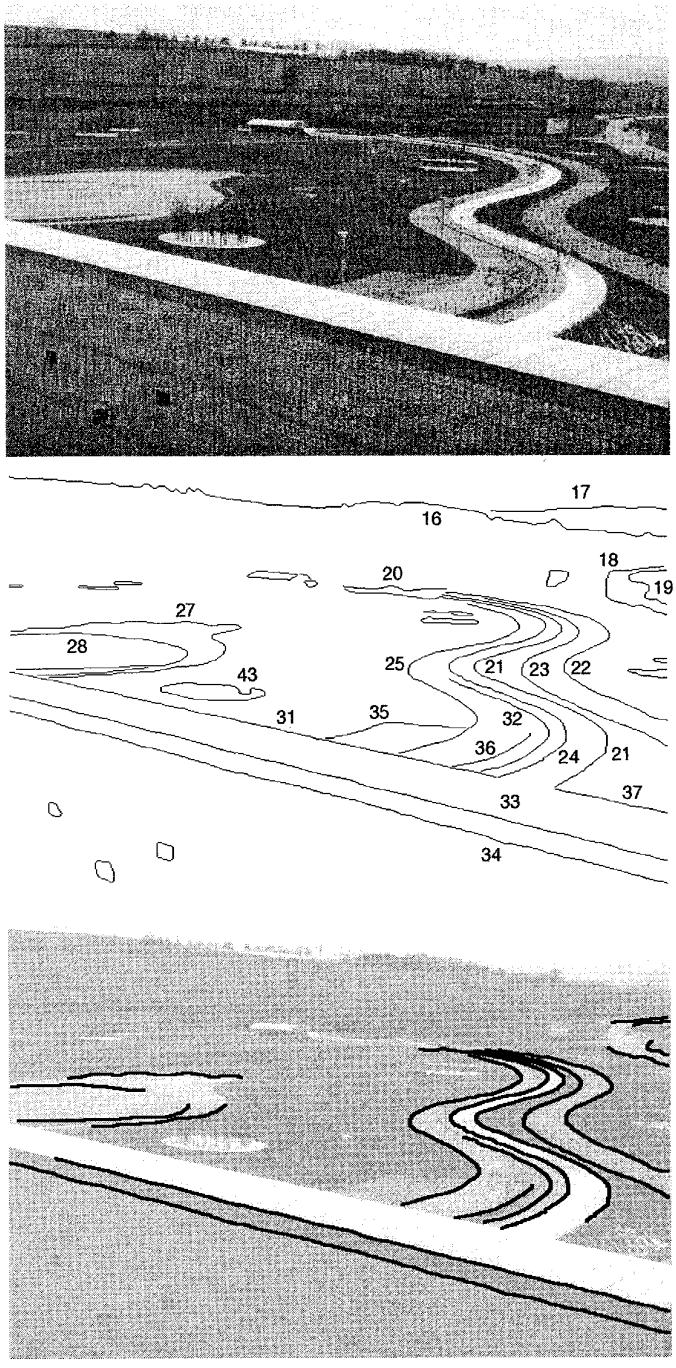

FIG. 13. An example of applying the algorithm to a country scene: (top) shows the original image; (middle) shows the hand traced edge map; (bottom) shows parallel curves detected in thick black curves. Greedy algorithm is used in determining the amount of shift in each iteration. Maximum number of iterations is limited to 10.

TABLE 1
Details on Parallelism Detection Results

Figure	Curve label	Curve size	Parallel with	Mapping scheme
11	18	73	24	38..73 → 23..51
	19	699	45	212..699 → 151..654
	21	59	23	0..59 → 20..80
	23	80	21	20..80 → 0..59
	24	51	18	23..51 → 38..73
	26	46	27	0..26 → 0..32
	27	117	45	0..117 → 122..0
	31	47	27	0..47 → 42..95
	36	93	19	0..82 → 644..569
	41	114	42	6..59 → 0..49
	42	49	41	0..49 → 6..59
	45	747	19	151..654 → 212..699
	46	366	45	0..366 → 0..382
	47	304	19	0..56 → 77..171; 66..170 → 204..353
	53	125	58	32..106 → 0..69
	58	193	53	0..69 → 32..106
	65	79	67	9..69 → 8..136
	67	154	65	8..136 → 9..69
	70	74	67	0..74 → 43..139
	72	46	45	0..46 → 657..602
12	16	159	19	40..159 → 12..94
	17	102	21	0..77 → 19..97
	19	105	16	12..94 → 40..159
	21	136	17	19..97 → 0.77
			25	109..136 → 0..26
	23	85	25	0..85 → 24..107
	24	95	28	4..52 → 23..71
	25	109	23	24..107 → 0..85
			21	0..26 → 109..136
	28	96	24	23..71 → 4..52
13	18	110	19	0..40 → 0..26; 65..110 → 52..75
	19	75	18	0..26 → 0..40; 52..75 → 65..110
	21	403	24	56..381 → 0..267
	22	253	23	39..253 → 0..241
	23	241	22	0..241 → 39..253
	24	300	21	0..267 → 56..381
	25	257	24	0..257 → 0..300
	27	350	28	43..173 → 0..99; 217..317 → 135..266
	28	266	27	0..99 → 43..173; 135..266 → 217..317
	32	138	24	10..133 → 179..300
	33	497	34	34..497 → 0..497
	34	497	33	0..497 → 34..497
	36	61	32	0..61 → 84..138

Note. See text for interpretations.

ods, Hough-based strategies or any transformation estimation methods described in [3, 6, 15, 24]. Once the underlying rotation is estimated, the expected tangents and orientation of the sleepers for the conjugate pairs are immediately available. Then, the whole scheme can be activated in the same manner. In other words, we can treat the problem as detecting parallelism undergoing rotation. The key issue then is to estimate the rotation parameters efficiently; this is under investigation by our group.

From all these discussions, several merits of the force-driven optimization technique can be drawn. First the method uses only local information and is therefore inherently parallel. Even when neighborhood information is needed in formulating the objective functions, only those results at the immediate neighbors are required. It is not difficult to visualize that the simplicity relies on forces that can counteract each other within an iteration and can propagate through the iterative process. More importantly, the mechanism allows an easy integration of several sources of information to obtain a solution, while keeping the decision

simple. Furthermore, run-time complexity increases linearly with the number of constraints involved. Consequently, adding new constraints or removing redundant criteria becomes very straightforward. Another merit of the technique is that it does not suffer from the uncontrollable flexibility as in variational calculus. Furthermore, singularity (e.g., in linear systems) does not exist in this approach, consequently an iteration is guaranteed to complete and the search direction for a better solution is always available.

There are also many aspects we can further improve upon. The most profitable direction is to employ multiresolution techniques to reduce the number of tests for long curves which have no parallelism in between. Another fruitful area of improvement is in the localized but parallel shifting strategy between iterations during the force-driven coupling process, which has a major effect on efficiency. Another direction for improving the algorithm is to derive a heuristic function to determine the order of testing for various combinations of simple segment endpoint coupling. This has been studied and reported in [32]. For future development, we shall apply the same strategy to detect other perceptual organizations for curves and to adopt the basic logic to shape matching, stereo matching, and motion tracking.

REFERENCES

1. A. Amini, T. Weymouth, and R. Jain, Using dynamic programming for solving variational problems in vision, *IEEE Trans. Pattern Anal. Mach. Intell.* **PAMI-12,** No. 9, 1990, 885–867.
2. W. Chan and F. Chin, ''Approximation of Polygonal Curves with Minimum Number of Line Segments or Minimum Error,'' Technical Report TR-93-04, Department of Computer Science, University of Hong Kong, 1993.
3. L. Chen and L. Davis, Parallel curve matching on the Connection Machine, *Pattern Recognit. Lett.* **14,** 1993, 133–140.
4. J. Dolan and R. Weiss, Perceptual grouping of curved lines, in *Proceedings, DARPA Image Understanding Workshop, 1989,* pp. 1135–1145.
5. R. Duda and P. Hart, *Pattern Classification and Scene Analysis,* Wiley, New York, 1973.
6. L. Gupta and M. Srinath, Invariant planar shape recognition using dynamic alignment, *Pattern Recognit.* **21,** 1988, 235–239.
7. D. Hoffman and W. Richards, Representing smooth plane curves for recognition, in *Proceedings, National Conference on Artificial Intelligence, 1982,* AAAI-82, pp. 5–8.
8. D. Huttenlocher and S. Ullman, ''Object Recognition using Alignment,'' *Proceedings, International Conference on Computer Vision, 1987,* pp. 102–111, 1987.
9. D. Jacobs, Robust and efficient detection of salient convex groups,'' *IEEE Trans. Pattern Anal. and Mach. Intell.* **PAMI-18,** No. 1, 1996, 23–37.
10. M. Kass, A. Witkin, and D. Terzopoulos, Snakes: active contour models, *Int. J. of Comp. Vis.,* 1988, 321–331.
11. K. Lai and R. Chin, ''On Regularization, Formulation and Initialization of the Active Contour Models (Snakes),'' *Proceedings, Asian Conference on Computer Vision, 1993,* pp. 542–545.
12. Z. Li, T. Bui, T. Tang, and C. Suen, *Computer Transformation of Digital Images and Patterns,* World Scientific, Singapore, 1989.
13. D. Lowe, *Perceptual Organisation and Visual Recognition,* Kluwer Academic, Boston, 1985.
14. D. Lowe, ''Three-Dimensional Object Recognition from Single Two-Dimensional Images,'' *Artif. Intell.* **31,** 1987, 335–395.
15. C. Lu and J. Dunham, Shape matching using polygon approximation and dynamic alignment, *Pattern Recognit. Lett.* **14,** 1993, 945–949.
16. R. Mohan, *Perceptual Organisation for Computer Vision, Ph.D. thesis, IRIS Technical Report 254,* University of Southern California, 1989.
17. R. Mohan and R. Nevatia, Using perceptual organisation to extract 3D structures, *IEEE Trans. Pattern Anal. Mach. Intell.* **PAMI-11,** No. 11, 1989, 1121–1139.
18. R. Mohan and R. Nevatia, Perceptual organisation for scene segmentation and description, *IEEE Trans. Pattern Anal. Mach. Intell.* **PAMI-14,** No. 6, 1992, 616–635.
19. G. Peters and J. Wilkinson, Eigenvalues of $Ax = \lambda Bx$ with band symmetric A and B, *Comp. J.* **12,** 1969, 398–404.
20. B. Ray and K. Ray, An algorithm for detecting of dominant points and polygonal approximation of digital curves, *Pattern Recognit. Lett.* **13,** 1992, 849–856.
21. P. Rosin, Multiscale representation and matching of curves using codons, *CVGIP: Graphical Models and Image Processing* **55,** No. 4, 1993, 286–310.
22. S. Sarkar and K. Boyer, Integration, inference, and management of spatial information using Bayesian networks: perceptual organization, *IEEE Trans. Pattern Anal. Mach. Intell.* **PAMI-15,** No. 3, 1993, 256–274.
23. S. Sarkar and K. Boyer, ''A Computational Structure for Preattentive Perceptual Organization: Graphical Enumeration and Voting Methods,'' *IEEE Trans. Syst. Man, Cybern.,* **24,** No. 2, 1994, 246–267.
24. J. Schwartz and M. Sharir, Identification of partially obscured objects in two dimensions by matching of noisy characteristic curves,'' *Int. J. Robotics Research* **6,** No. 2, 1987, 29–44.
25. F. Ulupinar and R. Nevatia, ''Inferring Shape from Contour for Curved Surfaces,'' *Proceedings in International Conference on Pattern Recognition, 1990,* pp. 147–154.
26. M. Wertheimer, Principles of perceptual organisation (translated), in *Readings in perception* (D. Beardslee and M. Wertheimer, Eds.), Van Nostrand, Princeton, NJ, 1958.
27. B. Widrow, The rubber mask technique, parts I and II, *Pattern Recognition* **5,** 1973, 175–211.
28. D. Williams and M. Shah, A fast algorithm for active contours and curvature estimation, *Computer Vision, Graphics, Image Processing,* **55,** 1992, 14–26.
29. A. Witkin and J. Tenenbaum, On the role of structure in vision, in *Human and Machine Vision* (J. Beck, B. Hope, and A. Rosenfeld, Eds.), pp. 481–543, Academic Press, New York, 1983.
30. W. H. Wong and H. S. Ip, Direction-dependent tangent: a new tangent representation, in *Proceedings, Image and Video Processing III,* **SPIE-2421,** 1995, 203–207.
31. W. H. Wong and H. S. Ip, ''Detecting Zero Curvature Points for Direction-Dependent Tangent on Discrete Curves,'' *Proceedings, Vision Geometry IV,* **SPIE-2573,** 1995, 172–182.
32. W. H. Wong and H. S. Ip, ''Heuristic Strategy for Feature Matching in Parallel Cures Detection,'' *Proceedings, 1995 IEEE International Conference on Systems, Man and Cybernetics,* **5,** 1995, 4273–4278.
33. W. H. Wong and H. S. Ip, On detecting parallel curves: models and representations, *Int. J. Pattern Recognit. Artif. Intell.* **10,** 1996, 813–827.
34. M. Worring and A. Smeulders, Digital curvature estimation, *CVGIP: Image Understanding* **58,** No. 3, 1993, 366–382.