Chapter 3

The Polyline Shape Search Problem

Shape comparison is a fundamental problem in computer vision because of the importance of shape information in performing recognition tasks. For example, many model-based object recognition algorithms work by matching boundary contours of imaged objects ([39, 73, 43, 44, 10, 55]). In addition to this traditional application, shape information is also one of the major components in some content-based image retrieval systems (see e.g. [51] and [46]). The goal in such a system is to find database images that look similar to a given query image or drawing. Images and queries are usually summarized by their color, shape, and texture content. For example, in the illustration retrieval system described in [11], we suggest a shape index which records *what* basic shapes (such as line segments, corners, and circular arcs) fit *where* in the drawing. The method in this chapter can be used to index shape information in images once contours have been extracted.

In this chapter, a shape is a polyline in the plane. We consider the following problem: Given two planar polylines, referred to as the *text* and the *pattern*, find all approximate occurrences of the pattern within the text, where such occurrences may be scaled and rotated versions of the pattern. We call this problem of locating a polyline pattern shape within a polyline text shape the *polyline shape search problem*, or PSSP for short. The PSSP is difficult because we allow both scaling and partial matching. Allowing scaling of the input pattern requires us to make precise certain inherent tradeoffs in PSSP matching. Consider for example, trying to match a line segment pattern into a text polyline. The pattern will match a segment of the text polyline exactly at a continuum of different scales. In this case, we only wish to report the maximum length match. In addition, suppose that the angle between two consecutive segments of the text is very close to 180°. In this case,

we may prefer a single longer match with a very small error that spans the two segments instead of two shorter, zero error, matches for each segment.

Two closely related problems to the PSSP are the *segment matching problem* ([40, 30, 31]) and the *polyline simplification problem* ([6, 24, 37, 49, 79]). The segment matching problem is to find approximate matches between a short polygonal arc and pieces of a longer polygonal arc. In searching for these matches, the short arc is allowed to rotate, but its length/scale remains constant. The PSSP generalizes the segment matching problem by allowing scaling. In the polyline simplification problem, a polyline and error bound are given, and we seek an approximation polyline with the fewest number of segments whose distance to the given polyline is within the error bound. (This problem is also known in the literature as the *min-#* problem.) For a polyline with n vertices, the planar polyline simplification problem can be solved in $O(n^2)$ time if the vertices of the approximation are required to be a subsequence of the vertices of the given polyline ([6]), and in $O(n)$ time if the vertices of the approximation can be arbitrary points in the plane but the given polyline is the graph of a piecewise linear function ([37]). One can think of the polyline simplification problem as an attempt to find long segments within the input polyline, i.e. the PSSP with a line segment pattern. The PSSP generalizes the polyline simplification problem because the PSSP allows for any polyline pattern, not just a line segment. PSSP matches are *not* required to start and end at vertices of the input text (and hence PSSP matches may overlap). For a text polyline with n vertices, our algorithm for the PSSP with a line segment pattern requires $O(n^2)$ time.

This chapter is organized as follows. In section 3.1, we describe the framework for our solution to the PSSP, including the match score for a given scale, orientation, and position of the pattern within the text. In section 3.2, we derive the orientation of the pattern which gives the best match for a fixed pattern scale and position. This leaves us with a 2D search problem in scale-position space (the position is the text arclength at which to begin the comparsion of the pattern with the text). In section 3.3, we show how a certain set of lines divides up the scale-position plane into regions in which we have analytic formulae for our scoring function. In section 3.4, we fully describe our line sweep algorithm for the PSSP. Section 3.5 shows some results of our algorithm. Finally, we summarize our work and suggest areas for future work in section 3.6.

The material in this chapter has been published in [13].

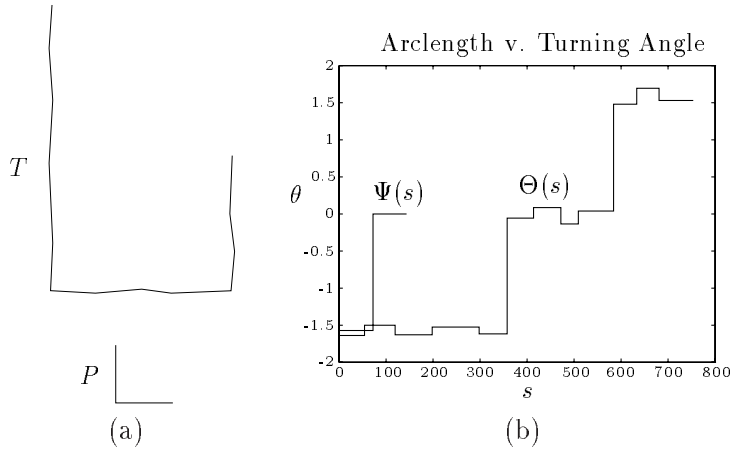


Figure 3.1: PSSP Turning Angle Summaries. (a) Text T above the corner pattern P. (b) Arclength versus cumulative turning angle functions $\Theta(s)$ and $\Psi(s)$ for T and P, respectively (s in points, θ in radians).

3.1 Problem Setup

Let T and P denote the text and pattern polylines, respectively. We will use the familiar arclength versus cumulative turning angle graph in judging the quality of a match. We denote these summary graphs for the text and pattern as $\Theta(s)$ and $\Psi(s)$, respectively. Figure 3.1 shows an example. If T consists of n segments and P consists of m segments, then $\Theta(s)$ and $\Psi(s)$ are piecewise constant functions with n and m pieces, respectively. We denote the text arclength breakpoints as $0 = c_0 < c_1 < \cdots < c_n = L$, where L is the length of the text. The value of $\Theta(s)$ over the interval (c_i, c_{i+1}) is denoted by θ_i, $i = 0, \ldots, n \Leftrightarrow 1$. Similarly, we denote the pattern arclength breakpoints as $0 = a_0 < a_1 < \cdots < a_m = l$, where l is the length of the pattern, and the value of $\Psi(s)$ over the interval (a_j, a_{j+1}) is ψ_j, $j = 0, \ldots, m \Leftrightarrow 1$.

Rotating the pattern by angle γ simply shifts its summary graph by γ along the turning angle axis, while scaling the pattern by a factor α stretches its summary graph by a factor of α along the arclength axis. Hereafter, a scaled, rotated version of the input pattern will be referred to as the *transformed pattern*. The comparison between the transformed pattern and the text will be done in the summary coordinate system. The text arclength at which to begin the comparison will be denoted by β. Since the length of the transformed pattern is αl, the transformed pattern summary graph is compared to the text summary graph from β to $\beta + \alpha l$. Finding the pattern within the text means finding a stretching, right shift, and up shift of the pattern summary graph $\Psi(s)$ that makes it closely resemble the corresponding

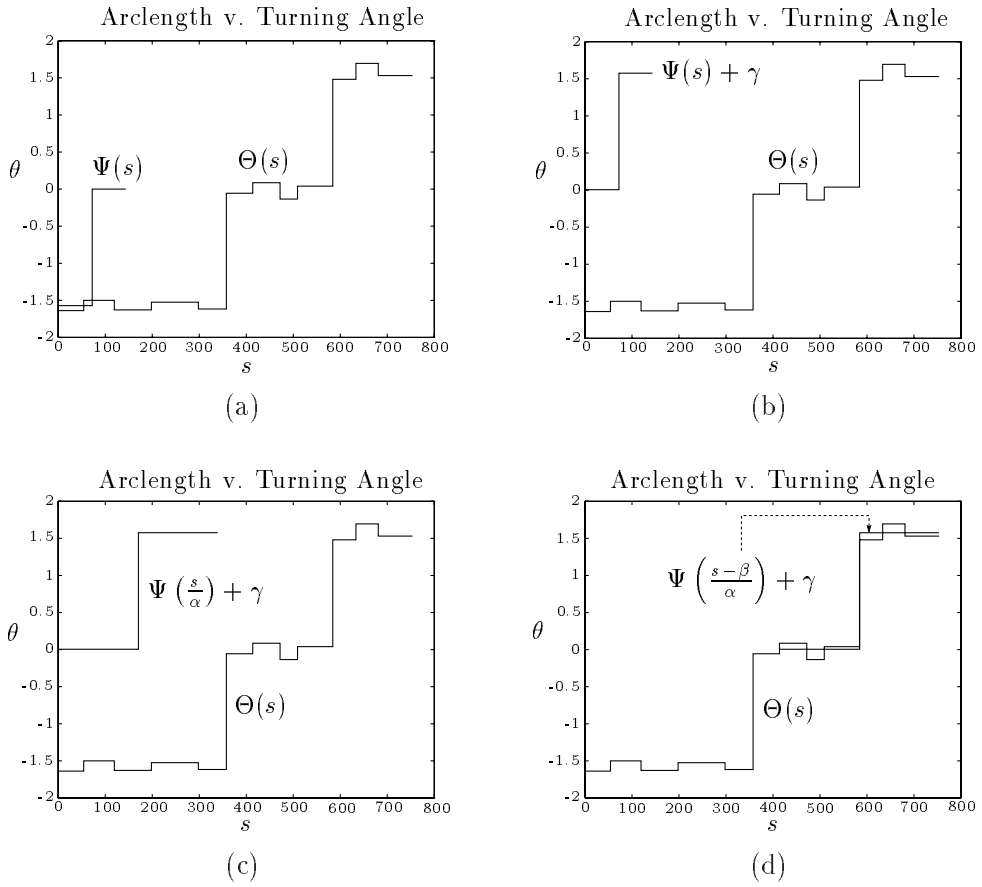


Figure 3.2: PSSP Matching in Arclength Versus Turning Angle Space. (a) $\Theta(s)$ and $\Psi(s)$ from Figure 1(b). (b) Rotating the pattern by γ shifts its summary graph up by γ. (c) Scaling the rotated pattern by α stretches the summary graph by a factor of α. (d) Finally, we slide the transformed pattern summary graph over by an amount β to obtain a good match.

piece of the text summary graph. Figure 3.2 illustrates this intuition. The stretching (α), up shifting (γ), and right shifting (β) of the pattern summary graph $\Psi(s)$ correspond to scaling, rotating, and sliding the pattern along the text, respectively. The problem of finding the pattern within the text is thus a search problem in the scale-position-rotation space (α, β, γ).

The preceeding discussion tacitly assumes that the pattern and text are both *open* polylines. If, for example, the text is closed (i.e. the text is a polygon), then the above strategy will miss a pattern match that crosses the arbitrary start and finish text arclengths $s = 0$ and $s = L$ which actually correspond to the same point on the text. Such a match will not be missed if we match the pattern summary curve to a text summary curve which

runs from $s = 0$ to $s = 2L$, where $\Theta(s + L) = \Theta(s) + 2\pi$ for $s \in [0, L)$. Since the underlying text summary is repeated twice, we need to be careful not to report two identical matches. In what follows, however, we ignore the issue of open/closed polylines, and simply assume that our pattern and text are open polylines.

In judging the quality of a match at a given scale, orientation, and position, we need to consider both the error of the match and the length of the match. Is a long match with a large error better than a short match with a small error? Using the mean squared error, for example, indicates an indifference to the length of the match. If the pattern were a single segment, then it would fit with zero mean squared error at a continuum of different scales and locations along each edge of the text polyline. The mean squared error metric cannot distinguish among these matches. In order to do so, we will use a match scoring function which rewards longer matches. When the mean squared error (length) of two matches is equal, the longer (lower mean squared error) match will have a higher score than the shorter (higher mean squared error) match. Of course, there is still the issue of how to compare a match to a shorter (longer), lower mean squared error (higher mean squared error) match. There is no one correct answer to this balancing question – the answer depends, for example, on the underlying input noise model. Here we opt for a simple balancing of match length versus match error which, as we shall see, yields very good results and is amenable to analysis via standard calculus optimization techniques. Obviously, other match score functions are possible.

In moving toward our scoring metric, we define the mean squared error $e(\alpha, \beta, \gamma)$ as

$$e(\alpha, \beta, \gamma) = \frac{\int_{s=\beta}^{\beta+\alpha l} \left(\Theta(s) \Leftrightarrow \left(\Psi\left(\frac{s-\beta}{\alpha}\right) + \gamma\right)\right)^2 \, ds}{\alpha l}.$$

Note that $\Psi((s \Leftrightarrow \beta)/\alpha) + \gamma$, $s \in [\beta, \beta + \alpha l]$, is the summary graph of the transformed pattern, starting at text arclength β. The score $S(\alpha, \beta, \gamma)$ of a match is defined in terms of the mean squared error as

$$S(\alpha, \beta, \gamma) = \frac{\alpha l}{L(1 + e(\alpha, \beta, \gamma))}.$$

The product αl is the length of the match (α, β, γ). Our goal is to find local maxima of the score function S over a suitable domain D. This domain is defined by restricting the values of α and β so that the domain of definition of the stretched, shifted pattern summary graph is completely contained in $[0, L]$ (which is the domain of definition of the text summary graph):

$$D = \{ \, (\alpha, \beta) \mid \alpha > 0, \ \beta \geq 0, \text{ and } \alpha l + \beta \leq L \, \}.$$

Although the range of the mean squared error e is $[0, \infty)$, the range of the score S over the domain D is $[0, 1]$. A match of length L with zero mean squared error receives the highest score of one. Instead of trying to locate local maxima of S in D, we will try to find local minima of its reciprocal

$$R(\alpha, \beta, \gamma) \equiv \frac{1}{S(\alpha, \beta, \gamma)} = \frac{L}{\alpha l}(1 + e(\alpha, \beta, \gamma)).$$

Note that the rotation γ affects only the mean squared error portion of the score.

At a local maximum location $(\alpha_*, \beta_*, \gamma_*)$ of S, a small change in pattern scale, orientation, or position within the text decreases the match score. We do not, however, want to report all local maxima because two very similar matches may be reported. We want to report a complete set of independent matches. By independent matches, we mean that any two reported matches should be significantly different in at least one of the defining components: scale, orientation, and position. If a pattern fits very well into a piece of text at a particular scale, orientation, and position within the text, then the pattern at the same scale and orientation but with a slightly different position will also fit very well. Both matches should not be reported. We report only those matches at which S is a local maximum in a topological neighborhood of the match, where the topological elements are the vertices, edges, and regions of an arrangement of a certain set of lines in scale-position space. For example, we output a vertex (α, β) of the arrangement only if it is a better match than its neighboring vertices and all local maximum locations on adjacent edges. (The complete algorithm is given in section 3.4.) Using a topological neighborhood instead of a geometric one is only a heuristic used to achieve our goal of reporting independent matches.

3.2 The Best Rotation

In this section we fix (α, β) and derive the rotation angle $\gamma = \gamma_*(\alpha, \beta)$ which minimizes the mean squared error $e(\alpha, \beta, \gamma)$ and, hence, the reciprocal match score $R(\alpha, \beta, \gamma)$. This is straightforward because e is differentiable with respect to γ:

$$\frac{\partial e}{\partial \gamma}(\alpha, \beta, \gamma) = 2\left(\gamma \Leftrightarrow \frac{\int_{s=\beta}^{\beta+\alpha l}\left(\Theta(s) \Leftrightarrow \Psi\left(\frac{s-\beta}{\alpha}\right)\right)\, ds}{\alpha l}\right).$$

The derivative $\partial e/\partial \gamma$ is equal to zero exactly when

$$\gamma = \gamma_*(\alpha, \beta) = \frac{\int_{s=\beta}^{\beta+\alpha l}\left(\Theta(s) \Leftrightarrow \Psi\left(\frac{s-\beta}{\alpha}\right)\right)\, ds}{\alpha l},$$

the mean value of the difference $\Theta \Leftrightarrow \Psi$ (more precisely, $\Theta(s) \Leftrightarrow \Psi((s \Leftrightarrow \beta)/\alpha)$) over the arclength interval of the match. Since $\partial^2 e/\partial\gamma^2(\alpha,\beta,\gamma) \equiv 2 > 0$, we conclude that for fixed α and β, the rotation angle that minimizes the mean squared error is $\gamma = \gamma_*(\alpha,\beta)$ given above. If we define $e_*(\alpha,\beta) \equiv e(\alpha,\beta,\gamma_*(\alpha,\beta))$, then

$$e_*(\alpha,\beta) = \frac{\int_{s=\beta}^{\beta+\alpha l} \left(\Theta(s) \Leftrightarrow \Psi\left(\frac{s-\beta}{\alpha}\right)\right)^2 \, ds}{\alpha l} \Leftrightarrow \left(\frac{\int_{s=\beta}^{\beta+\alpha l} \left(\Theta(s) \Leftrightarrow \Psi\left(\frac{s-\beta}{\alpha}\right)\right) \, ds}{\alpha l}\right)^2 .$$

The function $e_*(\alpha,\beta)$ is the variance of $\Theta \Leftrightarrow \Psi$ over the arclength interval of the match.

3.3 The 2D Search Problem

The result of the previous section allows us to eliminate the rotation parameter from consideration in our score and reciprocal score functions. We define $R_*(\alpha,\beta) = R(\alpha,\beta,\gamma_*(\alpha,\beta))$. Our goal now is to find local minima in the domain D of

$$R_*(\alpha,\beta) = \frac{L}{\alpha l}\left(1 + \frac{I_2(\alpha,\beta)}{\alpha l} \Leftrightarrow \left(\frac{I_1(\alpha,\beta)}{\alpha l}\right)^2\right), \tag{3.1}$$

where

$$I_1(\alpha,\beta) = \int_{s=\beta}^{\beta+\alpha l} \left(\Theta(s) \Leftrightarrow \Psi\left(\frac{s \Leftrightarrow \beta}{\alpha}\right)\right) \, ds \qquad \text{and} \tag{3.2}$$

$$I_2(\alpha,\beta) = \int_{s=\beta}^{\beta+\alpha l} \left(\Theta(s) \Leftrightarrow \Psi\left(\frac{s \Leftrightarrow \beta}{\alpha}\right)\right)^2 \, ds. \tag{3.3}$$

Consider the evaluation of the integral $I_1(\alpha,\beta)$ for a fixed pair (α,β). Since Θ and Ψ are piecewise constant functions, this integral can be reduced to a finite summation of terms such as the product of $(\theta_i \Leftrightarrow \psi_j)$ with the length of the overlap of the ith arclength interval (c_i, c_{i+1}) of $\Theta(s)$ and the jth arclength interval $(a_j\alpha + \beta, a_{j+1}\alpha + \beta)$ of $\Psi((s \Leftrightarrow \beta)/\alpha)$. In precise mathematical terms, we have

$$I_1(\alpha,\beta) = \sum_{i=0}^{n-1} \sum_{j=0}^{m-1} (\theta_i \Leftrightarrow \psi_j) \times X_{ij},$$

where $X_{ij} = |(c_i, c_{i+1}) \cap (a_j\alpha + \beta, a_{j+1}\alpha + \beta)|$ and $|J|$ is the length of interval J.

Let l_{ij} denote the line $a_j\alpha + \beta = c_i$, $i = 0,\ldots,n$, $j = 0,\ldots,m$, in the $\alpha\beta$-plane. The four lines

$$l_{ij} \quad : \quad a_j\alpha + \beta = c_i,$$

$$\begin{aligned} l_{i+1,j} &: a_j\alpha + \beta = c_{i+1}, \\ l_{i,j+1} &: a_{j+1}\alpha + \beta = c_i, \quad \text{and} \\ l_{i+1,j+1} &: a_{j+1}\alpha + \beta = c_{i+1} \end{aligned}$$

divide the $\alpha\beta$-plane into regions in which we may write down explicit analytic formulae for $X_{ij} = X_{ij}(\alpha,\beta) = |(c_i, c_{i+1}) \cap (a_j\alpha + \beta, a_{j+1}\alpha + \beta)|$. In each region, the formula for the size of the intersection is (at most) a degree one polynomial in α and β. For example, when $c_i < a_j\alpha + \beta < c_{i+1} < a_{j+1}\alpha + \beta$, it is easy to check that $X_{ij} = c_{i+1} \Leftrightarrow (a_j\alpha + \beta)$. This situation is depicted in Figure 3.3. Note that the given formula for X_{ij} also holds if we replace $<$ with $\leq$ in the comparisons between text and transformed pattern breakpoints. All possible formulae for $X_{ij}(\alpha,\beta)$ are illustrated in Figure 3.4. Now let $\mathcal{L}$ denote the set of $(n+1)(m+1)$ lines l_{ij} and let $\mathcal{A} = \mathcal{A}(\mathcal{L})$ denote the arrangement[1] in $\alpha\beta$-space of the lines in $\mathcal{L}$. In each face f of $\mathcal{A}$, we have a degree one polynomial formula for X_{ij}, $X_{ij}^f = u_{ij}^f\alpha + v_{ij}^f\beta + w_{ij}^f$. As explained above, the formula for $X_{ij} = X_{ij}(\alpha,\beta)$ is determined by the above–below relationship between (α,β) and each of the four lines $l_{ij}, l_{i+1,j}, l_{i,j+1}$, and $l_{i+1,j+1}$. From this fact, it is easy to see that the above–below relationship of (α,β) and the line l_{ij} affects only the four intersection formulae $X_{ij}, X_{i-1,j}, X_{i,j-1}$, and $X_{i-1,j-1}$.

Note that $\mathcal{A}$ is an arrangement of non-vertical lines. The slope of l_{ij} is $\Leftrightarrow a_j \leq 0$, so all lines have negative or zero slope. Note also that $\mathcal{A}$ is degenerate because there are many pairs of parallel lines $(l_{i_1,j} \parallel l_{i_2,j})$. An arrangement vertex v_{ijpq} is the intersection of l_{ij} and l_{pq}. The scaling and sliding $(\alpha,\beta) = v_{ijpq}$ of the pattern lines up exactly two pairs of breakpoints: $a_j\alpha + \beta$ coincides with c_i and $a_q\alpha + \beta$ coincides with c_p. An arrangement edge e is an open segment along some line l_{ij}. A scaling and sliding $(\alpha,\beta) \in e$ lines up exactly one pair of breakpoints: $a_j\alpha + \beta$ coincides with c_i. For an open arrangement face f, any scaling and sliding $(\alpha,\beta) \in f$ lines up no pairs of breakpoints.

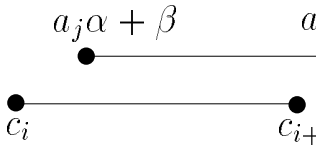
Now fix a face f of the arrangement $\mathcal{A}$ and let $Y_{ij} = \theta_i \Leftrightarrow \psi_j$. Then for (α,β) in the closure $\bar{f}$, the integrals (3.2), (3.3) in the formula for R_* can be written as

$$I_1^f(\alpha,\beta) = \sum_{i=0}^{n-1}\sum_{j=0}^{m-1} X_{ij}^f Y_{ij}, \qquad \text{and} \tag{3.4}$$

$$I_2^f(\alpha,\beta) = \sum_{i=0}^{n-1}\sum_{j=0}^{m-1} X_{ij}^f (Y_{ij})^2. \tag{3.5}$$

[1] For a survey of arrangements, see [19].

(a)



$$X_{ij} = c_{i+1} - (a_j\alpha + \beta)$$

$$\begin{array}{rcll} a_j\alpha+\beta & > & c_i & (\text{above } l_{ij}) \\ a_j\alpha+\beta & < & c_{i+1} & (\text{below } l_{i+1,j}) \\ a_{j+1}\alpha+\beta & > & c_i & (\text{above } l_{i,j+1}) \\ a_{j+1}\alpha+\beta & > & c_{i+1} & (\text{above } l_{i+1,j+1}) \end{array}$$

(b)

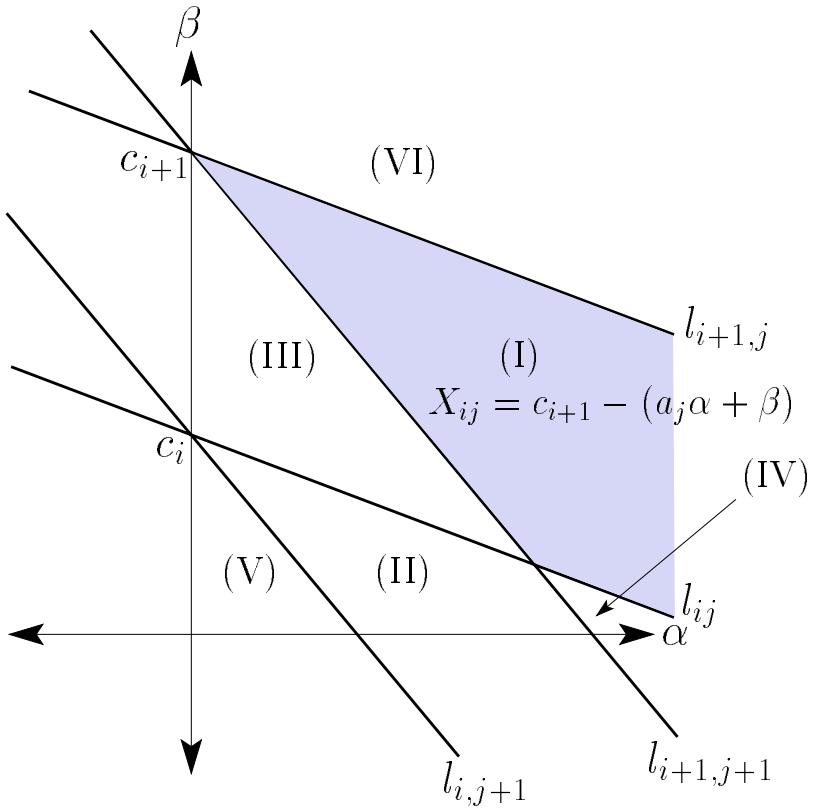


Figure 3.3: The Interval Overlap X_{ij}. (a) Intervals for which $c_i < a_j\alpha + \beta < c_{i+1} < a_{j+1}\alpha + \beta$. In this case, $X_{ij} = c_{i+1} \Leftrightarrow (a_j\alpha + \beta)$. (b) The corresponding region in scale-position space is shown in gray. The formulae for all six regions (I)–(VI) bounded by $\alpha \geq 0$, $\beta \geq 0$, l_{ij}, $l_{i+1,j}$, $l_{i,j+1}$, and $l_{i+1,j+1}$ are given in Figure 3.4.

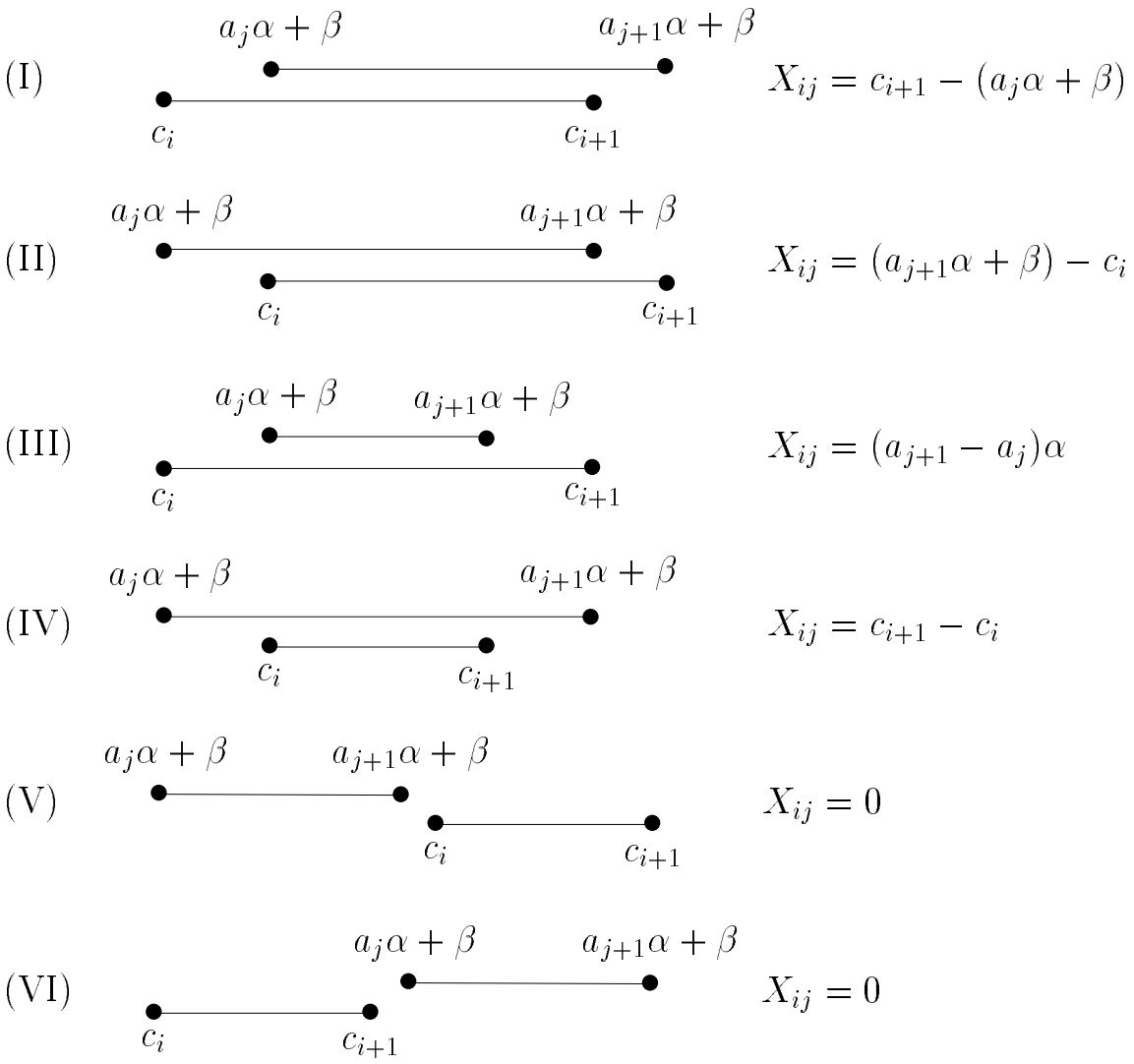


Figure 3.4: All Possible Interval Overlaps X_{ij}. In all cases, $X_{ij} = X_{ij}(\alpha, \beta)$ is (at most) a degree one polynomial in α and β. (I) $c_i \le a_j\alpha + \beta \le c_{i+1} \le a_{j+1}\alpha + \beta$. (II) $a_j\alpha + \beta \le c_i \le a_{j+1}\alpha + \beta \le c_{i+1}$. (III) $c_i \le a_j\alpha + \beta \le a_{j+1}\alpha + \beta \le c_{i+1}$. (IV) $a_j\alpha + \beta \le c_i \le c_{i+1} \le a_{j+1}\alpha + \beta$. (V) $a_j\alpha + \beta \le a_{j+1}\alpha + \beta \le c_i \le c_{i+1}$. (VI) $c_i \le c_{i+1} \le a_j\alpha + \beta \le a_{j+1}\alpha + \beta$.

Substituting $X_{ij}^f = u_{ij}^f \alpha + v_{ij}^f \beta + w_{ij}^f$ into these formulae and gathering like terms gives

$$I_1^f(\alpha,\beta) = \widetilde{u}^f \alpha + \widetilde{v}^f \beta + \widetilde{w}^f \qquad \text{and} \tag{3.6}$$

$$I_2^f(\alpha,\beta) = \widehat{u}^f \alpha + \widehat{v}^f \beta + \widehat{w}^f, \tag{3.7}$$

where $\widetilde{u}^f = \sum_{ij} u_{ij}^f Y_{ij}$, $\widehat{u}^f = \sum_{ij} u_{ij}^f (Y_{ij})^2$, and similarly for $\widetilde{v}^f$, $\widehat{v}^f$, $\widetilde{w}^f$, and $\widehat{w}^f$. Combining (3.1), (3.6), and (3.7), we can write R_* in the closed region $\bar{f}$ as

$$R_*^f(\alpha,\beta) = \frac{L}{\alpha^3 l^3}(A^f \alpha^2 + B^f \alpha\beta + C^f \beta^2 + D^f \alpha + E^f \beta + F^f) \tag{3.8}$$

for constants

$$A^f = l^2 + l\widehat{u}^f \Leftrightarrow (\widetilde{u}^f)^2,$$

$$B^f = l\widehat{v}^f \Leftrightarrow 2\widetilde{u}^f \widetilde{v}^f, \tag{3.9}$$

$$C^f = \Leftrightarrow (\widetilde{v}^f)^2, \tag{3.10}$$

$$D^f = l\widehat{w}^f \Leftrightarrow 2\widetilde{u}^f \widetilde{w}^f,$$

$$E^f = \Leftrightarrow 2\widetilde{v}^f \widetilde{w}^f, \qquad \text{and} \tag{3.11}$$

$$F^f = \Leftrightarrow (\widehat{w}^f)^2.$$

Our 2D search problem is to find pairs $(\alpha,\beta) \in D$ at which $R_*(\alpha,\beta)$ is a local minimum.

3.3.1 Faces

Can a local minimum of R_* occur in the interior of a face?[2] We have not been able to rule out this possibility, although we will argue that local minima inside faces are somewhat rare. We will also show that every open ball around a face local minimum location (α_*,β_*) contains other points (α,β) such that $R_*(\alpha,\beta) = R_*(\alpha_*,\beta_*)$. In this sense, there are no *strict* local minima of R_* inside an arrangement face. Furthermore, the same value $R_*(\alpha_*,\beta_*)$ can always be found on an arrangement edge or vertex which is adjacent to the face containing (α_*,β_*). Given the above considerations, we ignore the possibility of face local minima in our algorithm. In the rest of this section, we argue the previously made claims.

In order to determine if a local minimum of R_* can occur inside a face f, we must examine the function $R_*^f(\alpha,\beta)$ defined on $\bar{f}$ (see (3.8)).

[2] In [13], we concluded that there are no local minima of R_* inside an arrangement face. Our proof (given in the appendix of [13]), however, contained an error in the handling of the case $\widetilde{v}^f = 0$.

Theorem 1 *The function $R_*^f(\alpha,\beta)$ has no local minima in f if $\widetilde{v}^f \neq 0$ or $\widehat{v}^f \neq 0$. Note from equations (3.6) and (3.7) that $\widetilde{v}^f$ and $\widehat{v}^f$ are the coefficients of β in $I_1^f(\alpha,\beta)$ and $I_2^f(\alpha,\beta)$, respectively. Thus, the theorem can be rephrased to say that $R_*^f(\alpha,\beta)$ has no local minima if one of the integrals $I_1^f(\alpha,\beta)$ or $I_2^f(\alpha,\beta)$ depends on β.*

Proof. The first and second partial derivatives of R_*^f with respect to β are

$$\begin{aligned} \frac{\partial R_*^f}{\partial \beta} &= \frac{L}{\alpha^3 l^3}(B^f\alpha + 2C^f\beta + E^f) \\ \frac{\partial^2 R_*^f}{\partial \beta^2} &= \frac{2L}{\alpha^3 l^3}C^f. \end{aligned} \tag{3.12}$$

If $\widetilde{v}^f \neq 0$, then $C^f = \Leftrightarrow(\widetilde{v}^f)^2 < 0$, and, consequently, $\partial^2 R_*^f/\partial\beta^2 < 0$ (since $\alpha > 0$). The concavity of R_*^f with respect to β implies that $R_*^f(\alpha,\beta)$ cannot have a local minimum in the β direction for any α. Therefore, $R_*^f(\alpha,\beta)$ cannot have a local minimum when $\widetilde{v}^f \neq 0$.

So now suppose that $\widetilde{v}^f = 0$. From the formulae (3.9), (3.10), and (3.11), we see that $B^f = l\widehat{v}^f$, $C^f = 0$, and $E^f = 0$ in this case. The first derivative (3.12) then reduces to $\partial R_*^f/\partial\beta = (L/(\alpha^3 l^2))\widehat{v}^f$. Thus $\partial R_*^f/\partial\beta = 0$ iff $\widehat{v}^f = 0$. If follows that $R_*^f(\alpha,\beta)$ cannot have a local minimum when $\widetilde{v}^f = 0$ and $\widehat{v}^f \neq 0$.

We have shown that $R_*^f(\alpha,\beta)$ cannot have a local minimum when $\widetilde{v}^f \neq 0$ or ($\widetilde{v}^f = 0$ and $\widehat{v}^f \neq 0$). These conditions are, of course, logically equivalent to $\widetilde{v}^f \neq 0$ or $\widehat{v}^f \neq 0$. ■

Next, we seek an explicit condition on the text polyline which guarantees that $\widetilde{v}^f \neq 0$.

Define $i_0(\alpha,\beta)$ and $i_1(\alpha,\beta)$ to be the text indices where the transformed pattern graph domain $[\beta, \beta+\alpha l]$ begins and ends:

$$i_0(\alpha,\beta) = i_0 \text{ if } \beta \in [c_{i_0}, c_{i_0+1}) \qquad \text{and} \qquad i_1(\alpha,\beta) = i_1 \text{ if } \beta+\alpha l \in [c_{i_1}, c_{i_1+1}).$$

See Figure 3.5. For any $(\alpha,\beta) \in f$, no two breakpoints of the text and the transformed pattern graph line up. This means that $i_0(\alpha,\beta) \equiv i_0^f$ and $i_1(\alpha,\beta) \equiv i_1^f$ are constant for $(\alpha,\beta) \in f$. It also means that $I_1^f(\alpha,\beta)$ is differentiable with respect to β at points in f. By (3.2), $\partial I_1^f/\partial\beta = \partial/\partial\beta(\int_{s=\beta}^{\beta+\alpha l}\Theta(s)\,ds)$ since $\int_{s=\beta}^{\beta+\alpha l}\Psi((s \Leftrightarrow \beta)/\alpha)\,ds = \alpha\int_{s=0}^{l}\Psi(s)\,ds$ is independent of β. It is then easy to see from Figure 3.5 that $\partial I_1^f/\partial\beta = \theta_{i_1^f} \Leftrightarrow \theta_{i_0^f}$. But from (3.6), we also have that $\partial I_1^f/\partial\beta = \widetilde{v}^f$. Therefore $\widetilde{v}^f = \theta_{i_1^f} \Leftrightarrow \theta_{i_0^f}$ and $\widetilde{v}^f \neq 0$ iff $\theta_{i_1^f} \neq \theta_{i_0^f}$. Combining this fact and Theorem 1, we get

Corollary 1 *If $\theta_{i_0^f} \neq \theta_{i_1^f}$, then R_*^f has no local minima in f.*

The previous discussion does not eliminate the possibility of a local minimum of R_*^f if $i_0^f = i_1^f$. Fortunately, we have the following lemma.

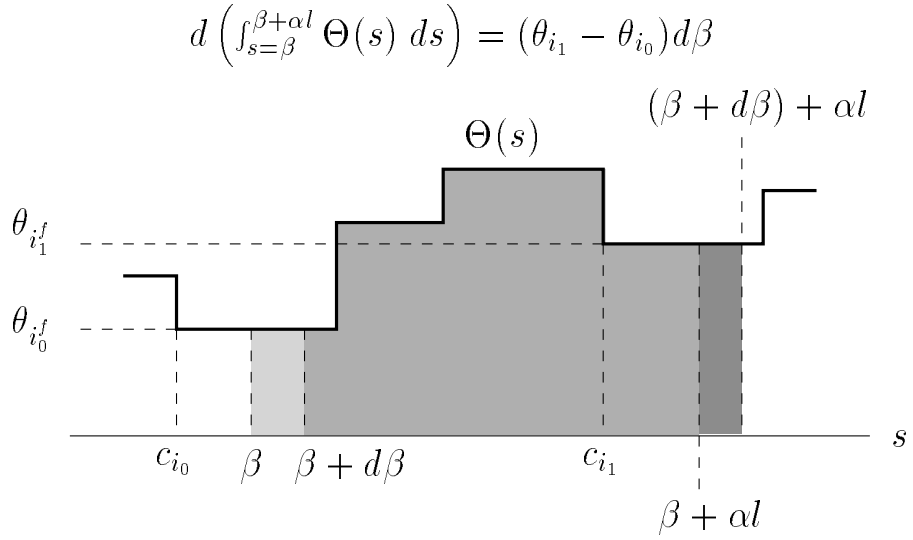


Figure 3.5: The Integral of Text Angle as a Function of Pattern Placement. Consider the integral $I(\beta) = \int_{s=\beta}^{\beta+\alpha l}\Theta(s)\,ds$, where α is fixed. Clearly, $I(\beta+d\beta) = I(\beta) \Leftrightarrow \int_{s=\beta}^{\beta+d\beta}\Theta(s)\,ds + \int_{s=\beta+\alpha l}^{\beta+d\beta+\alpha l}\Theta(s)\,ds$. If (α,β) is in the open face f, then we can choose $d\beta > 0$ small enough so that $(\alpha,\beta+d\beta)$ is still in f, $\Theta(\beta) = \Theta(\beta+d\beta) = \theta_{i_0^f}$, and $\Theta(\beta+\alpha l) = \Theta(\beta+d\beta+\alpha l) = \theta_{i_1^f}$. Here, we have $I(\beta+d\beta) \Leftrightarrow I(\beta) = (\theta_{i_1^f} \Leftrightarrow \theta_{i_0^f})d\beta$.

Lemma 1 *If $i_0^f = i_1^f$, then R_*^f has no local minima in f unless $R_*^f \equiv 0$.*

Proof. If $i_0^f = i_1^f$, then the text turning angle graph $\Theta(s) \equiv \theta_{i_0}$ over $[\beta, \beta+\alpha l]$ for every pair $(\alpha,\beta) \in f$. From equations (3.2) and (3.3), we can explicitly compute $I_1^f(\alpha,\beta)$ and $I_2^f(\alpha,\beta)$. The results are $I_1^f(\alpha,\beta) = \alpha l(\theta_{i_0^f} \Leftrightarrow \overline{\psi})$ and $I_2^f(\alpha,\beta) = \alpha l(\theta_{i_0^f}^2 \Leftrightarrow 2\theta_{i_0^f}\overline{\psi} + \overline{\psi^2})$, where $\overline{\psi} = (\int_{s=0}^{l}\Psi(s)\,ds)/l$ and $\overline{\psi^2} = (\int_{s=0}^{l}\Psi^2(s)\,ds)/l$. Substituting these results into equation (3.1) for R_*^f, we get $R_*^f(\alpha,\beta) = K/\alpha$, where K is constant with respect to α. If $K = 0$, then $R_*^f \equiv 0$. If $K \neq 0$, then $\partial R_*^f/\partial\alpha = \Leftrightarrow K/\alpha^2 \neq 0$, and hence again R_*^f has no local minima. ■

The following theorem combines Corollary 1 and Lemma 1.

Theorem 2 *If all text cumulative angles θ_i are distinct, then R_*^f has no local minima in f unless $R_*^f \equiv 0$.*

Proof. Consider the face f. If $i_0^f = i_1^f$, then the conclusion follows from Lemma 1. If $i_0^f \neq i_1^f$, then $\theta_{i_0^f} \neq \theta_{i_1^f}$ by assumption, and the conclusion follows from Corollary 1. ■

An almost immediate consequence of Theorem 2 is

Corollary 2 *If the text contains only right turns or only left turns (e.g. if the text is a piece of a convex polygon), then R_*^f has no local minima in f unless $R_*^f \equiv 0$.*

Proof. If the text only has right turns, then $\theta_0 < \theta_1 < \cdots < \theta_{m-1}$; if the text only has left turns, then $\theta_0 > \theta_1 > \cdots > \theta_{m-1}$. In either case, the cumulative turning angles θ_i are distinct, and the result follows from Theorem 2. ■

By Theorem 1, if there is a local minimum in f, we must have $\widetilde{v}^f = \widehat{v}^f = 0$. In this case, R_*^f reduces to $R_*^f(\alpha, \beta) = (L/(\alpha^3 L^3))(A^f\alpha^2 + D^f\alpha + F^f)$, which is independent of β. In order to determine if there is a local minimum in f, we minimize this function with respect to α. Here $\partial R_*^f/\partial\alpha = (\Leftrightarrow L/(\alpha^4 l^3))(A^f\alpha^2 + 2D^f\alpha + 3F^f)$, so there are at most two values of α at which a local minimum can occur. Obviously, these values can be determined in constant time. Even if R_*^f has a local minimum in the right halfspace $H = \{\ (\alpha, \beta)\ :\ \alpha > 0\ \}$, it may not occur in the open face f. Although we have not been able to eliminate the possibility of a local minimum inside a face, we have the following theorem.

Theorem 3 *There are no strict local minima of R_* in the interior of an arrangement face.*

By a *strict* local minimum at $(\alpha_*, \beta_*) \in f$, we mean that $R_*^f(\alpha_*, \beta_*) < R_*^f(\alpha, \beta)$ for all $(\alpha, \beta) \neq (\alpha_*, \beta_*)$ in a small enough open ball centered at (α_*, β_*) and contained within f. Theorem 3 follows from the fact that a local minimum in f implies that R_*^f is independent of β (as argued above). If (α_*, β_*) is the location of such a local minimum, then $R_*^f(\alpha_*, \beta_*) = R_*^f(\alpha_*, \beta)\ \forall(\alpha_*, \beta) \in \bar{f}$. In particular, $R_*^f(\alpha_*, \beta_*) = R_*^f(\alpha_*, \beta)$ if $(\alpha_*, \beta) \in \partial f$. Therefore, a small value $R_*^f(\alpha_*, \beta_*)$ will also occur on the edges and/or vertices of f which intersect the vertical line $\alpha = \alpha_*$.

Let us summarize the results of this section. We have not able to eliminate the possibility of a local minimum inside an arrangement face. However, such a local minimum cannot occur unless $\widetilde{v}^f = \widehat{v}^f = 0$ (Theorem 1), a fairly restrictive condition which implies that R_*^f does not depend on the shift β. Even if there were a local minimum of R_* at $(\alpha_*, \beta_*) \in f$, $R_*(\alpha_*, \beta_*)$ will *not* be strictly smaller than R_* at other points in a neighborhood of (α_*, β_*) (Theorem 3). Furthermore, the value $R_*^f(\alpha_*, \beta_*)$ will also occur on some arrangement edge or vertex adjacent to f. Since local minima inside faces are somewhat rare, are never strict local minima, and the same value can always be found on arrangement edges or vertices, we ignore the possibility of face local minima in our algorithm.

3.3.2 Edges and Vertices

Now consider an edge e that bounds face f. We want to know whether R_* has a local minimum at some $(\alpha, \beta) \in e$. For simplicity of computation, we check instead whether R_* has a local minimum at some $(\alpha, \beta) \in e$ *along the direction of* e. This is a weaker condition than R_* having a local minimum at (α, β), and it is this weaker condition that defines an

"edge minimum" in the algorithm presented in the next section. The edge e is part of a line $l_{ij} : a_j\alpha + \beta = c_i$. Combining this line equation with the equation (3.8) for R^f_*, we get a function R^e_* (R_* restricted to edge e) which is a rational cubic in α (the numerator is quadratic, but the denominator is cubic):

$$R^e_*(\alpha) \;=\; \frac{L}{\alpha^3 l^3}(\eta^f_{ij}\alpha^2 + \xi^f_{ij}\alpha + \rho^f_{ij}) \qquad \text{if } e \subseteq l_{ij},\, e \subseteq \partial f,$$

where

$$\begin{aligned}
\eta^f_{ij} &= A^f \Leftrightarrow a_j B^f + a_j^2 C^f \;, \\
\xi^f_{ij} &= c_i B^f \Leftrightarrow 2a_j c_i C^f + D^f \Leftrightarrow a_j E^f \;, \text{ and} \\
\rho^f_{ij} &= c_i^2 C^f + c_i E^f + F^f \;.
\end{aligned}$$

Thus we are left with a basic optimization problem. There are at most two local minima points $(\alpha_*, \beta_*) \in e$ for R^e_* since

$$\frac{dR^e_*}{d\alpha} = \Leftrightarrow\frac{L}{\alpha^4 l^3}(\eta^f_{ij}\alpha^2 + 2\xi^f_{ij}\alpha + 3\rho^f_{ij}).$$

These edge local minima locations can be determined in constant time given the formula for R^f_*.

The function R_* consists of piecewise rational cubic patches glued together at arrangement edges. Local minima of R_* may occur at arrangement vertices, so we must also examine the values of R_* at these locations. In fact, in practice we have found that values at vertices are smaller than minima on incident edges, even with the weaker notion of edge minimum given above. We shall say a bit more about this at the beginning of the results section 3.5.

3.4 The Algorithm

The user specifies a minimum match length $\text{matchlen}_{\min}$ and a maximum match length $\text{matchlen}_{\max}$ (default maximum is L), along with a bound on the maximum mean absolute error $\text{mae}_{\max}$ of a reported match. It is a bit easier to think in terms of the mean absolute error than in terms of the mean squared error since the former is in units of radians (or degrees), while the latter is in units of radians squared. If we let $f(s) = |\Theta(s) \Leftrightarrow(\Psi((s \Leftrightarrow \beta)/\alpha) + \gamma)|$, then the mean absolute error is $\text{mae}(\alpha, \beta, \gamma) = < f, I > /(\alpha l)$, where $< f, g >= \int_{s=\beta}^{\beta+\alpha l} f(s)g(s)\, ds$ and $I(s) \equiv 1$. The mean squared error is $\text{mse}(\alpha, \beta, \gamma) = ||f||^2/(\alpha l)$, where $||f||^2 =< f, f >$. Applying the Cauchy-Schwarz inequality gives $(< f, I >)^2 \leq ||f||^2||I||^2 =$

$||f||^2(\alpha l)$, from which it follows that $\text{mae}^2(\alpha,\beta,\gamma) \leq \text{mse}(\alpha,\beta,\gamma)$. Therefore, the bound $\text{mae}_{\max}$ can be guaranteed as long as we require the reported match to have a mean squared error which is less than or equal to $\text{mse}_{\max} = \text{mae}^2_{\max}$.

We say that (α,β) is *admissible* if the match length $\alpha l \in [\text{matchlen}_{\min}, \text{matchlen}_{\max}]$ and the mean squared error $e_*(\alpha,\beta) \leq \text{mse}_{\max}$. Note that the mean squared error $e_* = \frac{\alpha l}{L} R_* \Leftrightarrow 1$, and so we can quickly determine the mean squared error from the reciprocal score R_*. Of all admissible locations (α,β), we report only those which are locally the best. An admissible vertex is reported iff its reciprocal score is less than the reciprocal scores of all adjacent admissible vertices and of all admissible edge minima locations on adjacent edges. An admissible edge minimum is reported iff its reciprocal score is less than the reciprocal scores of all its admissible vertices (at most two) and the other admissible edge minimum (if one exists) on the same edge. Checking only a constant number of topologically neighboring elements does not guarantee that two very similar matches (which are geometrically close in $\alpha\beta$-space) will not be reported. Using topological closeness instead of geometric closeness is only a heuristic.

Our algorithm outputs matches during a topological sweep ([21]) over the $O(mn)$ lines l_{ij} in the degenerate arrangement $\mathcal{A}$. Our sweep implementation uses Edelsbrunner's "Simulation of Simplicity" technique ([20]) to cope with the degenerate input. During an elementary step, the topological sweep line moves from a face f_1 into a face f_2 through an elementary step vertex v. Please refer to Figure 3.6. During this step, we first compute the formula

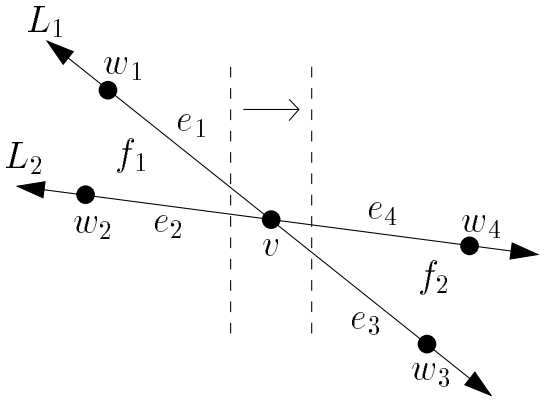


Figure 3.6: Elementary Step Notation.

for $R_*^{f_2}(\alpha,\beta)$. This requires computing the coefficients $\widetilde{u}^{f_2}$, $\widetilde{v}^{f_2}$, $\widetilde{w}^{f_2}$, $\widehat{u}^{f_2}$, $\widehat{v}^{f_2}$, and $\widehat{w}^{f_2}$ in the formulae (3.6) and (3.7) for the integrals $I_1(\alpha,\beta)$ and $I_2(\alpha,\beta)$, $(\alpha,\beta) \in \bar{f_2}$. Note that computing and summing the $O(mn)$ terms in (3.4) and (3.5) during each elementary

step would require total time $O(m^3n^3)$ because there are $O(m^2n^2)$ elementary steps. Fortunately, only a constant number of terms in (3.4) and (3.5) change when we move from face f_1 to face f_2. This is because only a constant number (at most eight) of intersection formulae X_{ij}^f are affected by above–below relationships involving L_1 and L_2. Hence, the values $\widetilde{u}^{f_2}, \widetilde{v}^{f_2}, \widetilde{w}^{f_2}, \widehat{u}^{f_2}, \widehat{v}^{f_2}, \widehat{w}^{f_2}$ in (3.6) and (3.7) can be computed in constant time from the values $\widetilde{u}^{f_1}, \widetilde{v}^{f_1}, \widetilde{w}^{f_1}, \widehat{u}^{f_1}, \widehat{v}^{f_1}, \widehat{w}^{f_1}$, and the formula for $R_*^{f_2}(\alpha, \beta)$ can be computed in $O(1)$ time from the formula for $R_*^{f_1}(\alpha, \beta)$. The latter formula was computed when the sweep line first entered face f_1.

The remaining work during an elementary step over v is straightforward. In Figure 3.6, e_3 and e_4 are the arrangement edges with v as left endpoint. These edges are part of the lines L_1 and L_2, respectively, and have right endpoints w_3 and w_4, respectively. From the formula for $R_*^{f_2}(\alpha, \beta)$, we compute the formulae for $R_*^{L_1}(\alpha, \beta)$ and $R_*^{L_2}(\alpha, \beta)$, as well as the values $R_*(v), R_*(w_3), R_*(w_4)$. From the formulae for $R_*^{L_1}(\alpha, \beta)$ and $R_*^{L_2}(\alpha, \beta)$, we compute the locations on e_3 and e_4 of any local minima of R_* in the direction of these edges. The above computations take constant time given the formula for $R_*^{f_2}(\alpha, \beta)$. During the elementary step at v, we decide whether to output the vertex v and any local edge minima locations on e_3 and e_4. During previous elementary steps, it was determined if it is still possible to report v by comparing the value of R_* at v to the value of R_* at w_1, w_2, and at all edge minima locations on e_1 and e_2. The remaining values of R_* needed to decide whether or not to report v are computed during the elementary step over v, as described above. During this step, we also compute all values of R_* needed to decide whether to report any local edge minima locations on e_3 and e_4. For each location (α, β) to be reported, we compute $\gamma = \gamma_*(\alpha, \beta) = I_1(\alpha, \beta)/\alpha l$ and report the triple (α, β, γ).

The above discussion shows that the elementary step work specific to our setting may be performed in $O(1)$ time. Thus, the total time to perform the topological sweep over the $O(mn)$ lines l_{ij} is $O(m^2n^2)$. The total space required by our algorithm is the $O(mn)$ storage required by a generic topological line sweep which does not store the discovered arrangement. In the common case of a simple pattern, such as a line segment or corner, $m = O(1)$ and our algorithm requires $O(n^2)$ time and $O(n)$ space.

The α, γ components of a reported triple (α, β, γ) give the scaling and rotation components of a similarity transformation of the pattern which makes it look like a piece of the text. The β component tells us where along the text this similar piece is located. To get the translation parameters of the similarity transformation, we sample the transformed pattern and the corresponding similar piece of the text, and find the translation parameters which minimize the mean squared error between the translated pattern point set and the text point set.

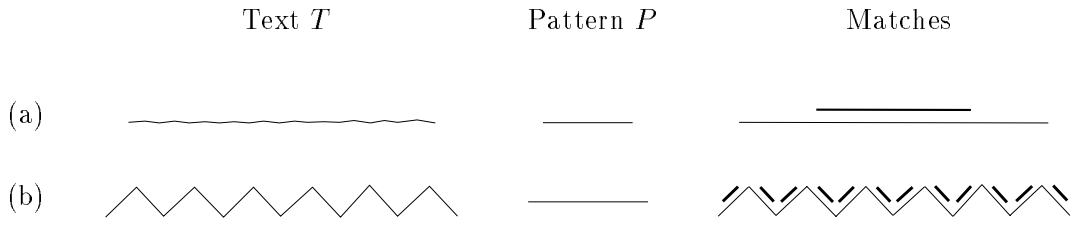


Figure 3.7: Trading Off Match Length Against Match Error. In both examples, the pattern is a line segment and the maximum absolute error input parameter $\text{mae}_{\max} = 9°$. To help make individual matches clear, we show a darker, smaller scale version of the pattern slightly offset from each match. (a) One match over the length of the entire "noisy" straight line text is found. (b) Twelve matches, one for each of the sides of the "mountain range", are found.

3.5 Results

In practice, local edge minima are very rarely reported because there is almost always a smaller admissible minimum at one of the two edge vertices. Essentially, the algorithm reports admissible vertices which have a reciprocal score which is lower than any other adjacent admissible vertex. Recall that arrangement vertices (α, β) give scalings and shifts of the pattern which cause two of its arclength breakpoints to line up with two of the text arclength breakpoints. Our experimentation has thus showed that the best matches of arclength versus turning angle graphs are usually those that line up two pairs of breakpoints (as opposed to one pair for points on arrangement edges and zero pairs for points in arrangement faces).

Figure 3.7 shows two experiments that illustrate the balancing of match error versus match length. In both cases, the pattern shape is a line segment. In the first case, the text is a "noisy" straight line. The angles between consecutive segments are all nearly 180°. With $\text{mae}_{\max} = 9°$, our algorithm finds one match over the length of the entire piece of text. In the second case, the text looks more like a mountain range. The angles between consecutive segments are far from flat angles. With $\text{mae}_{\max} = 9°$, our algorithm finds twelve matches, one for each of the sides of the mountain range.

A successful PSSP algorithm must produce a complete set of independent matches for given user tolerances. Figures 3.8 shows the result of applying our PSSP algorithm in two "exact" situations. In the first example, the pattern is simply a rotated version of the text. In the second example, the pattern is a rotated, scaled-down version of a portion of the text. For both inputs, only the one correct match is reported. Figure 3.9 shows the output of our

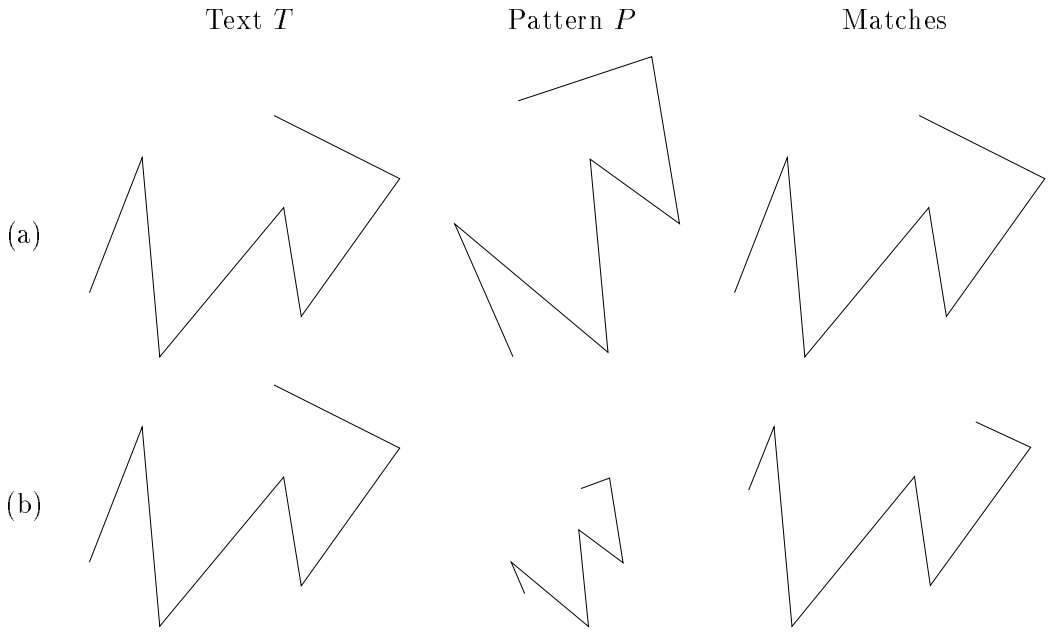


Figure 3.8: PSSP Exact Matching Examples. The columns (from left to right) show the text, pattern, and matches found. With $\text{mae}_{\max} = 9°$, our algorithm finds only the one exact match in both examples. (a) The pattern is a rotated version of the text. (b) The pattern is a rotated, scaled version of a piece of the text.

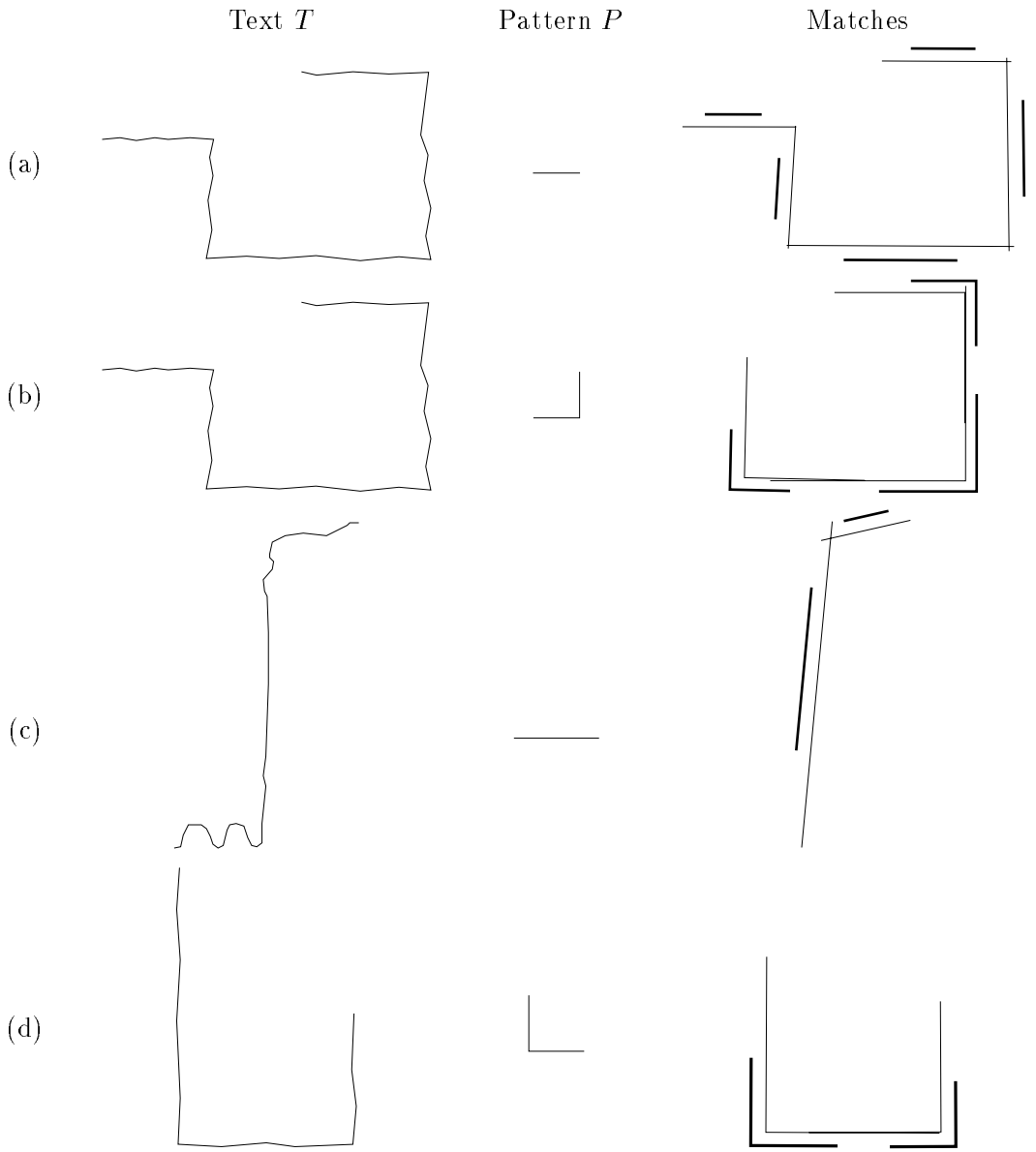


Figure 3.9: PSSP Results. As in Figure 3.7, each match is accompanied by a darker, smaller scale version of the pattern which is slightly offset from the match. (a) $\text{mae}_{\max} = 15°$. Each of the five noisy lines gives rise to exactly one segment match. (b) $\text{mae}_{\max} = 9°$. The three left turn text corners are found with the left turn corner pattern. (c) $\text{mae}_{\max} = 20°$. Our algorithm finds the two (relatively) long, straight portions of the text. (d) $\text{mae}_{\max} = 9°$. Both left turn text corners are found.

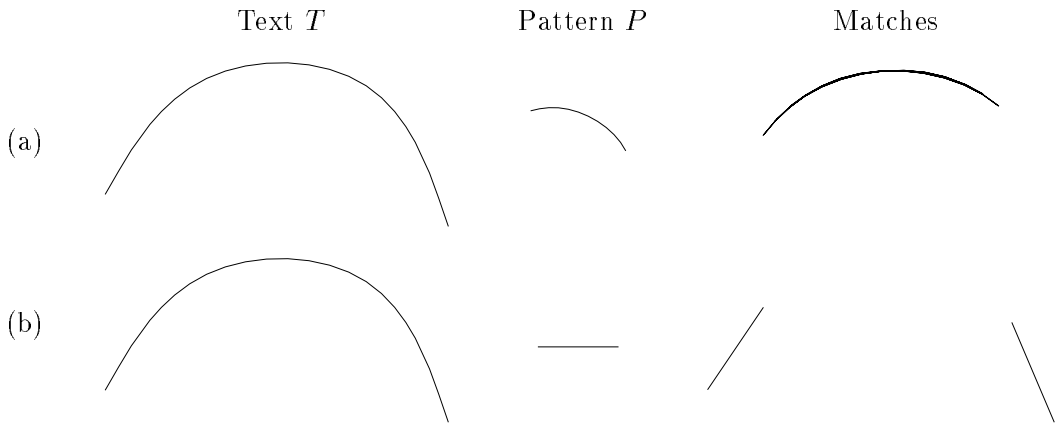


Figure 3.10: More PSSP Results. In both examples, the text is a curve with straight line and circular portions. (a) The circular arc pattern only matches the circular part of the text. (b) The line segment pattern only matches the straight parts of the text.

algorithm on four (text, pattern) pairs in which the pattern is either a straight segment or a corner. In Figure 3.9(b) we clearly see that the order of the vertices in the pattern and text makes a difference in the matches found by our algorithm — the three left turn text corners are found, but the right turn text corner is missed. To get around this dependence on the representation of the polyline, we could run our algorithm again on the pattern represented by vertices in the reverse order (so that the pattern is a right turn instead of a left turn). In Figure 3.10, we fit a circular arc and straight line pattern into a text curve with both circular and straight portions. The circular arc does not match the straight portions of the text, nor does the line segment match the circular portion of the text. In the examples shown in Figures 3.11 and 3.12, we use our method to summarize the straight line content of an image.

Obviously, if the pattern shape is not present anywhere within the text (within user specified tolerances) then a PSSP algorithm should not report any matches. Given the text and pattern shown in Figure 3.13 our PSSP algorithm reports no matches for $\text{mae}_{\text{max}} = 9°$ and $\text{matchlen}_{\text{min}}$ set to prevent very tiny matches.

3.6 Summary and Suggestions for Future Work

In this chapter we developed an algorithm to find where a planar "pattern" polyline fits well into a planar "text" polyline. By allowing the pattern to rotate and scale, we find

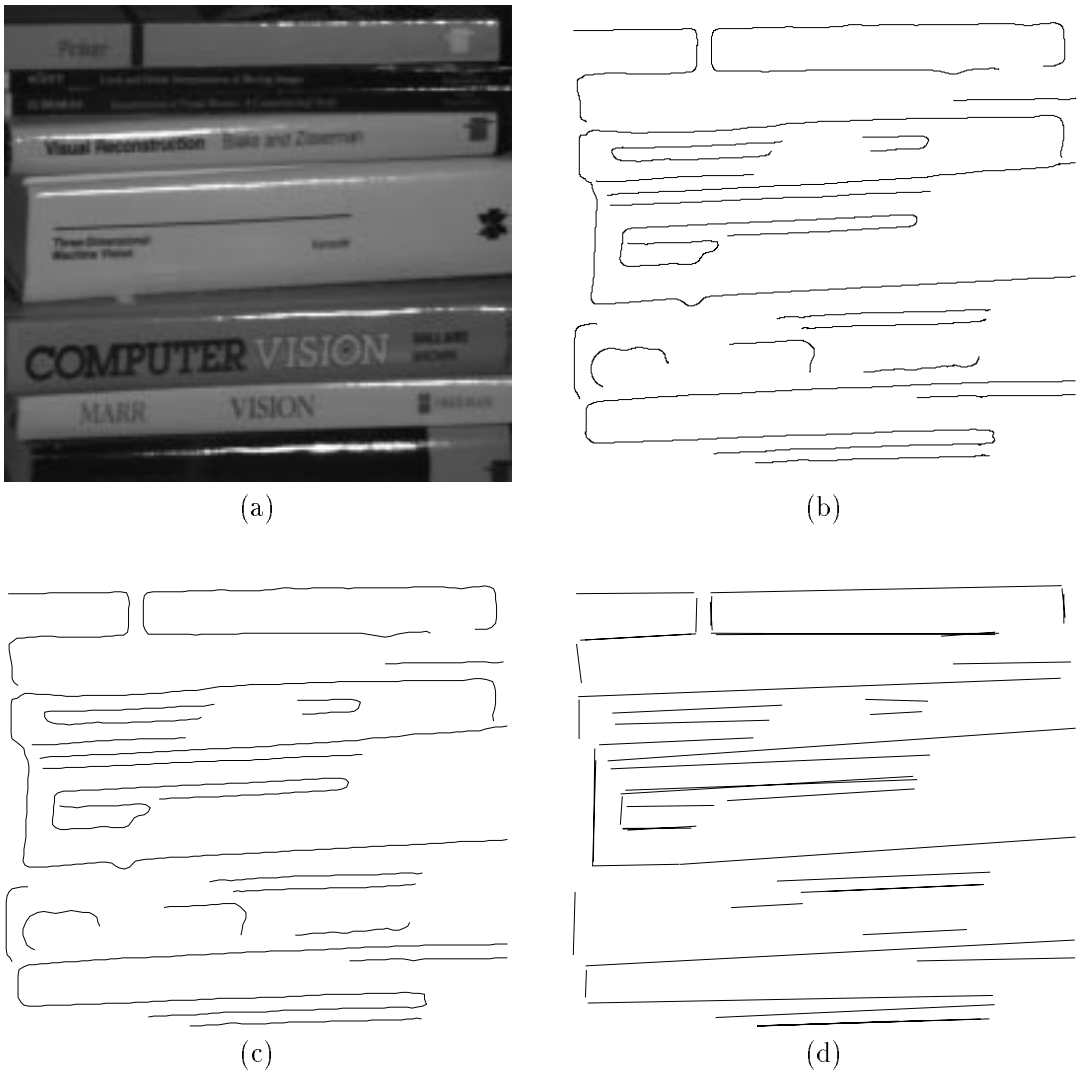


Figure 3.11: Image Summary by Straight Segments. (a) The image to be summarized is 512×480. (b) The result of Canny edge detection ($\sigma = 6$ pixels) and edgel linking is a set of polylines with a total of 7219 vertices. (c) The result of subsampling each of the polylines by a factor of 6 leaves a total of 1212 vertices. (d) Finally, fitting a straight segment to each of the subsampled polylines using our PSSP algorithm gives a set of 50 segments. As mentioned in the text, checking only a constant number of topologically neighboring elements before reporting a match (α, β) does not guarantee that two very similar matches (which are geometrically close) will not be reported. This heuristic is responsible for the "double edges" in the image summary.

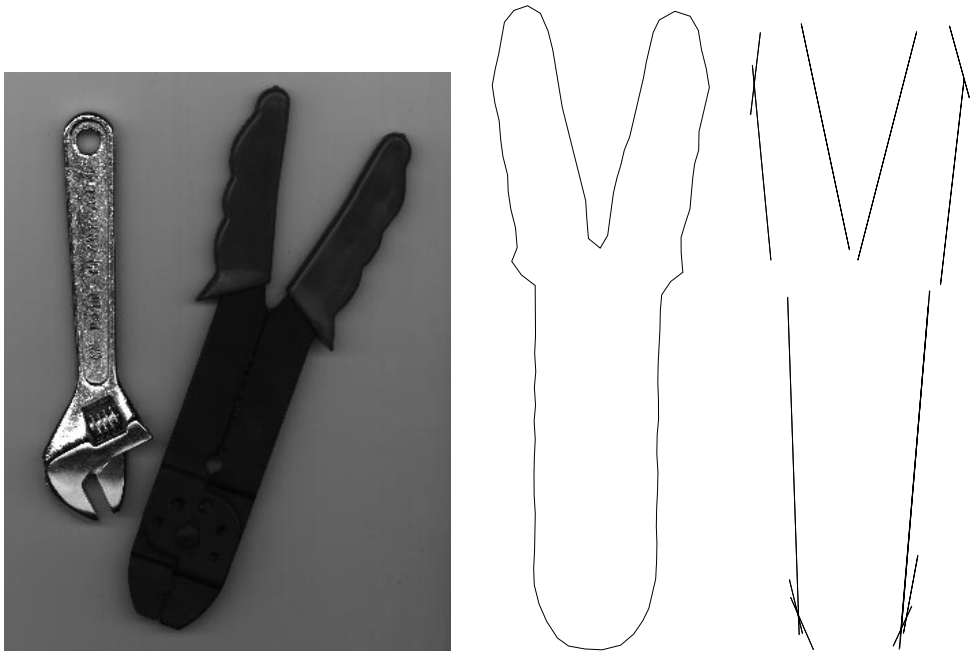


Figure 3.12: Another Image Summary by Straight Segments. (a) The "tools" image. (b) The result of Canny edge detection and edgel linking of the pliers' contour. (c) The result of fitting a straight segment into the pliers' contour using our PSSP algorithm.

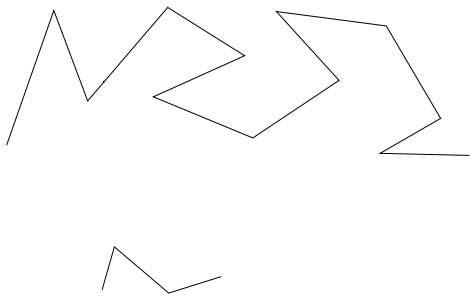


Figure 3.13: PSSP No Matches Example. The text is shown above the pattern. Our PSSP algorithm reports no matches for $\text{mae}_{\text{max}} = 9°$ and $\text{matchlen}_{\text{min}}$ set to prevent very tiny matches.

portions of the text which are similar in shape to the pattern. We dealt with the issue of match length versus match error by using a match scoring function that balances these two factors. All comparisons were performed on the arclength versus cumulative turning angle representations of the polylines. This allowed us to reduce the complexity of the problem: To compare two planar polylines, we compare their one-dimensional arclength versus turning angle graphs. In addition, the space operations of scaling, rotation, and sliding the pattern along the text cause simple changes to the pattern summary graph. Another reduction in complexity was gained by using the L_2^2 distance to compare the graphs. This allowed us to eliminate the rotation parameter from the search space, leaving a 2D scale-position space. Thus, we converted a four-dimensional search problem in the space of similarity transformations to a two-dimensional search problem in scale-position space.

Our line sweep strategy essentially examines all possible pattern scales and positions within the text. If, however, the pattern does not fit well at a certain scale and location, then it will not fit well at nearby scales and locations. Finding "certificates of dissimilarity" which would allow us to prune our (α, β) search space is a topic for future research. Another idea for speeding up our algorithm is to use the results of pattern searches in coarse versions of the text (with relatively high error tolerances) to guide searches in finer versions of the text, with the final search in the text itself. This hierarchical search strategy aims to reduce the total search time by reducing the number of expensive comparisons between the pattern and the full text at the expense of many cheaper comparisons with coarser versions of the text. The two techniques discussed above will help speed up the search for one pattern within one text. Suppose, however, that we have multiple patterns that we want to fit into multiple texts. This is the case, for example, if we are trying to summarize an image by recording which basic shapes/patterns fit where in the image. Can we do better than the brute force approach of applying our algorithm to each (text, pattern) pair?

The problem considered in this chapter can be generalized in a few different ways. For example, we may want to find affinely transformed versions of a pattern within a text. In addition to the shape transformation group, we could expand the class of shapes to include planar shapes other than polylines, such as circular arcs or cubic splines. Also, we may want to remove the restriction that the one-dimensional shapes are planar and allow, for example, polylines in three dimensions. To complete the generalization of the problem, we could allow the two-dimensional shapes such as polyhedra or spheres. Another possible topic of further research is to use the ideas in this chapter to develop a metric on polyline shapes which returns a small distance between two polylines when a scaled, rotated version of one matches well a portion of the other. See [3][34][54][2] for work on shape metrics.

One strategy for handling other one-dimensional planar pattern and text shapes is to

apply our method on polyline approximations of the input shapes. An accurate polyline approximation, however, may require a very large number of segments and our algorithm uses $O(m^2n^2)$ time. In choosing the arclength versus turning angle representation of a shape, we restricted ourselves to one-dimensional planar shapes. In addition, our algorithm relies heavily on the simple changes to this representation when the underlying polyline is uniformly scaled and rotated. This no longer holds if we allow, for example, affine transformations of the pattern.

Although our algorithm does not generalize to handle more general shape search problems, it produces excellent results for planar polylines under similarity transformations. This very specific problem is far from trivial because we match possibly scaled versions of the pattern to pieces of the text, and we require only the "best" matches to be reported. The main idea of the algorithm is to divide up the search plane into regions in which it is (relatively) easy to handle a complex scoring function. This general idea is obviously applicable to other search problems.