Notes for studying "Approximate Matching of Polygonal Shapes"

Section One – Introduction

I. <u>Abstract:</u>

For two given simple polygons *P*, *Q* the problem is to <u>determine a rigid motion *I* of *Q* giving the best possible match between *P* and *Q*, i.e. minimizing the Hausdorff-distance between *P* and *I*(*Q*).</u> Faster algorithms as the one for the general problem are obtained for special cases, namely that *I* is restricted to translations or even to translations only in one specified direction. It turns out that determining pseudo-optimal solutions, i.e. ones that differ from the optimum by just a constant factor can be done much more efficiently than determining optimal solutions. In the most general case the algorithm for the pseudo-optimal solution is based on the quite surprising fact that for the optimal possible match between *P* and an image *I*(*Q*) of *Q*, **the distance between the centroids of the edges of the convex hulls of *P* and *I*(*Q*) is a constant multiple of the Hausdorff-distance between *P* and *I*(*Q*).** It is also shown that the Hausdorff-distance between two simple polygons can be determined in time *O*(*n* log *n*), where *n* is the total number of vertices.

II. <u>Problem:</u>

Given *P*, *Q*, find an isometry *I* such that the distance between *P* and *I*(*Q*) is minimized. Furthermore , determine that minimal distance.

III. <u>Input:</u> Two polygons *P* and *Q*
<u>Output:</u> An isometry *I* such that $\delta_H(P, I(Q))$ is minimal

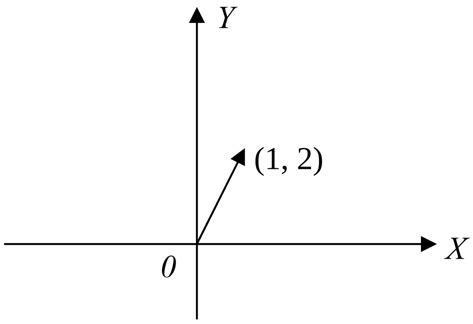
IV. An algebraic idea: ***isometry***:

An ***isometry*** is an affine mapping in the plane which preserves distances.
Any isometry I can be represented as $I = r \circ \rho \circ t$, where *r* is the reflexion at the x-axis, ρ is a rotation about the origin and t is a translation (it slides every point the same distance in the same direction [pp. 118 at "*A First Course in Abstract Algebra*, 6ed., John B. Franleigh"]).

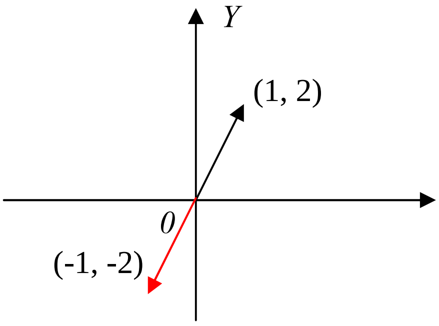
V. ***Rigid motions*** or ***even isometries*** are isometries without reflexion (= reflection).

VI. Any isometry is of the form: $I(x) = M \cdot x + t$, where $M = \begin{pmatrix} \cos\psi & \sin\psi \\ -\sin\psi & \cos\psi \end{pmatrix}$ for some $\psi \in [0, 2\pi]$ and $t \in R^2$ is some fixed translation vector.

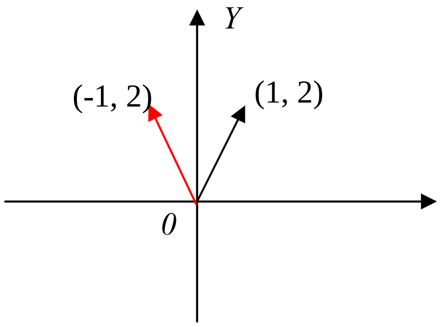
For example,



$$\begin{bmatrix} \cos\pi & \sin\pi \\ -\sin\pi & \cos\pi \end{bmatrix} \begin{bmatrix} 1 \\ 2 \end{bmatrix} = \begin{bmatrix} -1 & 0 \\ 0 & -1 \end{bmatrix} \begin{bmatrix} 1 \\ 2 \end{bmatrix} = \begin{bmatrix} -1 \\ -2 \end{bmatrix}$$



$$\begin{bmatrix} \cos(-53^\circ) & \sin(-53^\circ) \\ -\sin(-53^\circ) & \cos(-53^\circ) \end{bmatrix} \begin{bmatrix} 1 \\ 2 \end{bmatrix} = \begin{bmatrix} \frac{3}{5} & -\frac{4}{5} \\ \frac{4}{5} & \frac{3}{5} \end{bmatrix} \begin{bmatrix} 1 \\ 2 \end{bmatrix} = \begin{bmatrix} -1 \\ 2 \end{bmatrix}$$



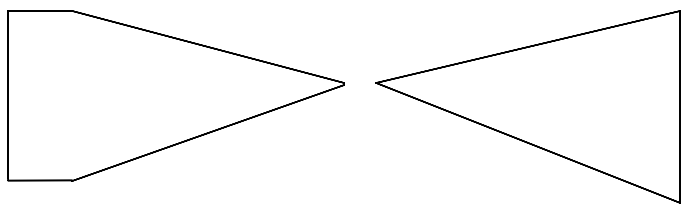
Actually, these properties can be seen from *linear algebra*.

VII. Such problems like this have some special cases

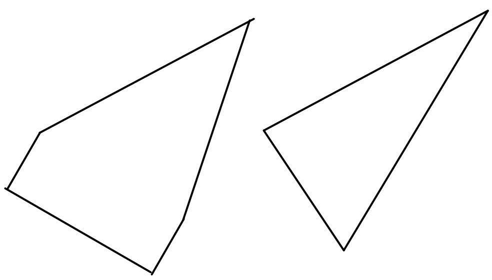
> P_3: The general problem: $I(x) = M \cdot x + t$
>
> P_0: No Isometries (except for the identity): $I(x) = M \cdot x + t$, where M is the identity matrix *Id*, $t = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$.
>
> P_1: Only translations along one fixed direction t_0 are allowed: $I(x) = M \cdot x + t$, where $M = Id$ and $t \in \{\lambda \cdot t_0 \mid \lambda \in \mathbf{R}\}$.
>
> P_2: Only translations are allowed: $I(x) = M \cdot x + t$, where $M = Id$.

Section Two – Determine the Hausdorff-distance of Two Polygons & Problem P_1

I. We should notice that any two polygons may have close points, yet, they cannot be considered as being close. For example,



The two polygons above have close points, but we don't want to consider these two polygons to be close. However, in the following case,



These two polygons are closer. How to determine that if two polygons are closer or not, that is, how close they are? The concept of Hausdorff-distance can satisfy our expectancy.

II. ***Hausdorff-metric*** δ_H is defined by:

$\delta_H(A,B)=\max(\tilde{\delta}_H(A,B),\tilde{\delta}_H(B,A))$, where $\tilde{\delta}_H(X,Y)=\sup_{x\in X}\inf_{y\in Y} d(x,y)$ is the distance from X to Y and $d(x, y)$ is the Euclidean distance in the plane. $\delta_H(A,B)$ is always defined if $A,B\subset R^2$ are bounded. Note that "inf" = infimum and "sup" = superior.

III. Consider the following example. Assume that $X = \{x_1, x_2\} = \{(0, 1), (1, 0)\}$, $Y = \{y_1, y_2\} = \{(1, 2), (2, 3)\}$.

$$\begin{aligned}\tilde{\delta}_H(X,Y)=\sup_{x\in X}\inf_{y\in Y} d(x,y)&=\sup\{\inf\{d(x_1,y_1),d(x_1,y_2)\},\inf\{d(x_2,y_1),d(x_2,y_2)\}\}\\ &=\sup\{\inf\{\sqrt{2},2\sqrt{2}\},\inf\{2,\sqrt{10}\}\}\\ &=\sup\{\sqrt{2},2\}\\ &=2.\end{aligned}$$

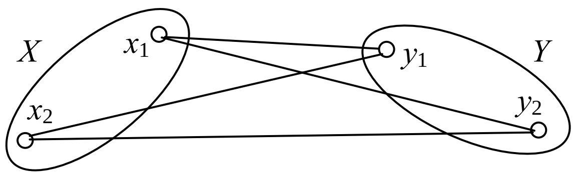
What if we try to calculate $\delta_H(Y,X)$?

$$\begin{aligned}\tilde{\delta}_H(Y,X)=\sup_{y\in Y}\inf_{x\in X} d(y,x)&=\sup\{\inf\{d(y_1,x_1),d(y_1,x_2)\},\inf\{d(y_2,x_1),d(y_2,x_2)\}\}\\ &=\sup\{\inf\{\sqrt{2},\sqrt{2}\},\inf\{2\sqrt{2},\sqrt{10}\}\}\\ &=\sup\{\sqrt{2},2\sqrt{2}\}\\ &=2\sqrt{2}.\end{aligned}$$

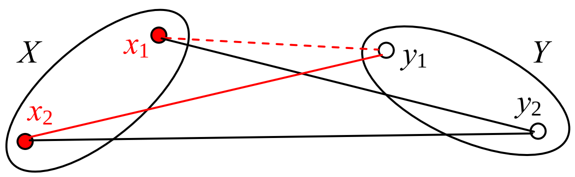
We find that $\tilde{\delta}_H(Y,X)\neq\tilde{\delta}_H(X,Y)$ in this case.

Therefore, we obtain that $\delta_H(X,Y)=\max(\tilde{\delta}_H(X,Y),\tilde{\delta}_H(Y,X))=2\sqrt{2}.$

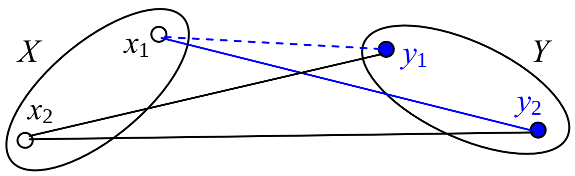
Let's see what $\tilde{\delta}_H(X,Y)$ and $\tilde{\delta}_H(X,Y)$ present in R^2 domain by the following graph:



For X, $\tilde{\delta}_H(X,Y)$ can be calculated by the following figure:

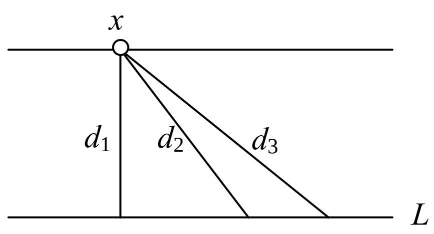


For Y, $\tilde{\delta}_H(Y,X)$ can be calculated by the following figure:



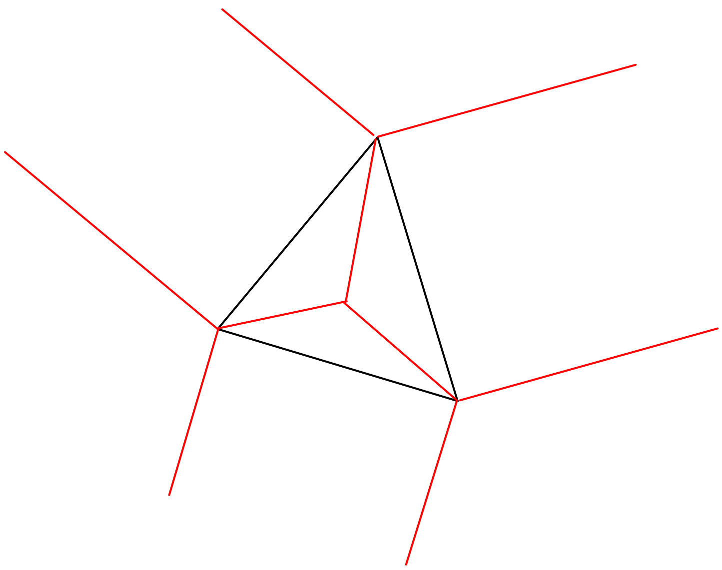
We can roughly say that $\tilde{\delta}_H(X,Y)$ is the longest distance starting from X to Y and $\tilde{\delta}_H(X,Y)$ is the longest distance from Y to X.

Actually, the part " $\inf_{y \in Y} d(x,y)$ " of " $\tilde{\delta}_H(X,Y) = \sup_{x \in X} \inf_{y \in Y} d(x,y)$ " represent the real distance. This concept can be easily understood from the below figure:



As we can see from the above figure, the *distance* from x to the line L is d_1.

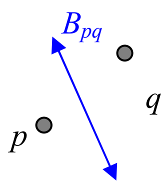
IV. In order to solve problem P_0, i.e., determine the Hausdorff-distance between P and Q, we consider the Voronoi diagram of P. We denote the Voronoi diagram of P Vor(P). A Voronoi diagram of a polygon P is as follows: (I take a triangular for an example. I want to thank Prof. Hwang here for his help to solve some problems.)



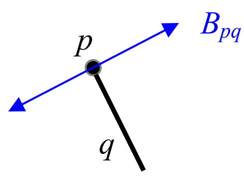
How do I get it? Principles of drawing a Voronoi diagram of a polygon come from the following paper:

[F87] **A Sweepline Algorithm for Voronoi Diagrams**, Forrune, S., **Algorithmica**, Vol. 2, 1987, pp. 153-174.

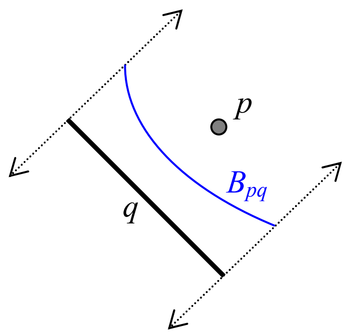
Case 1:



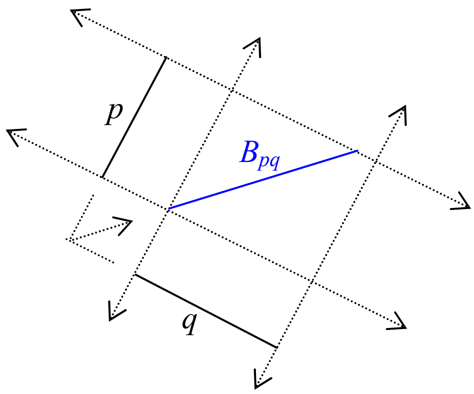
Case 2:



Case 3:



Case 4:



V. The following papers proposed a linear time algorithm to find the Hausdorff-distance between two polygons:

[A83] A Linear Time Algorithm for the Hausdorff Distance between Convex Polygons, Atallah, M. J., **Information Processing Letters**, Vol. 17, 1983, pp. 207-209.

VI. ***How Voronoi Diagrams help to Find Hausdorff-distance***?

(a) Lemma 1:

> The distance of polygon Q to polygon P, $\tilde{\delta}_H(Q,P)$ is assumed either at some vertex of Q or at some intersection point of Q with some Voronoi-edge e of P having either the smallest or largest x-coordinate among the intersection points of Q with e.

For example,

Assume that we have two polygons P ad Q as Fig. 1:

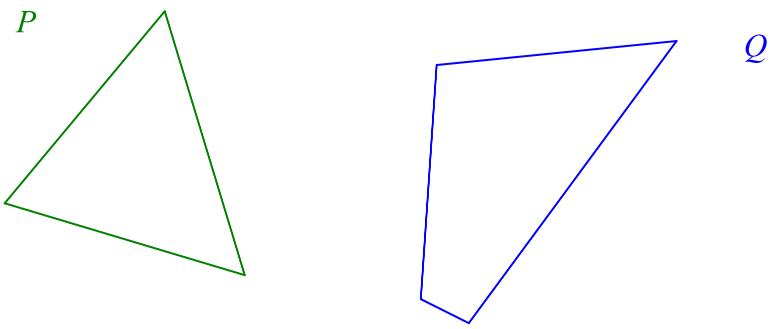


Fig. 2-1: Two polygons P and Q

In this case, we just consider the isometry $I = t$, where t is a fixed translation. Then we get P and $I(Q)$ and obtain $\tilde{\delta}_H(I(Q),P) = h$ as Fig. 2.

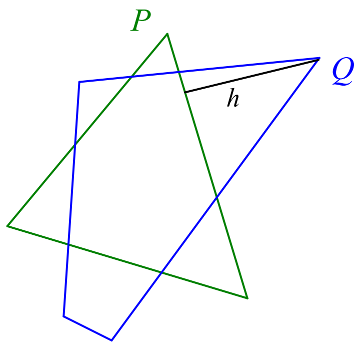


Fig. 2-2: P, $I(Q)$ and $\tilde{\delta}_H(I(Q), P)$

Now, we draw a Voronoi diagram for P as Fig. 3,

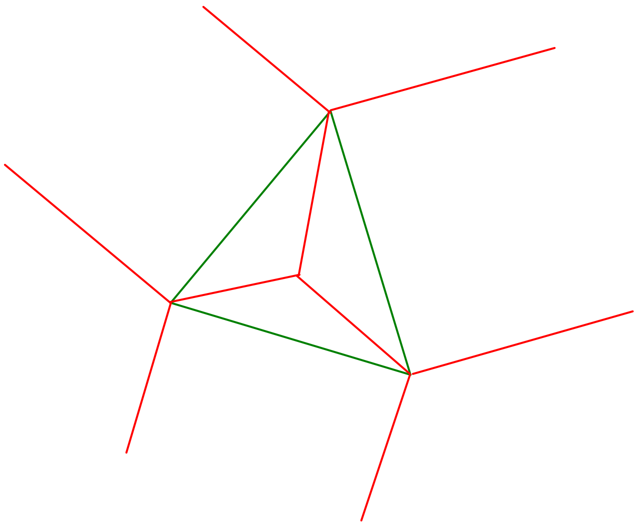


Fig. 2-3: Voronoi diagram for polygon P

And then we obtain that

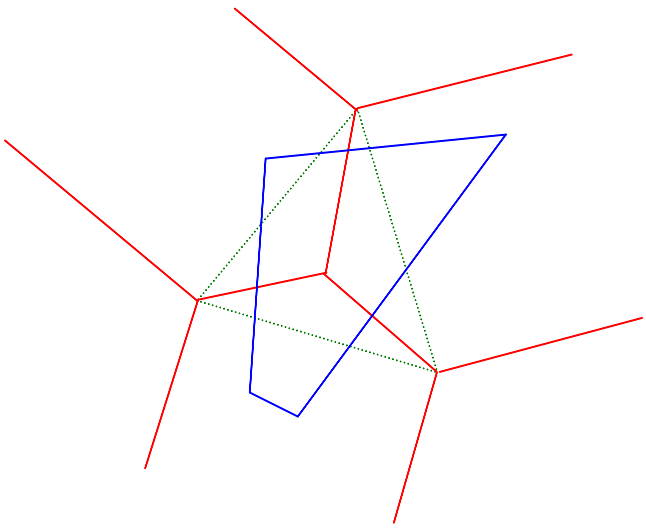


Fig. 2-4: Vor(P) intersects Q.

Consider the intersection of a fixed Voronoi-cell C within Q. Suppose we move monotonically on an edge e of Q within this Voronoi-cell C as Fig. 5 shows.

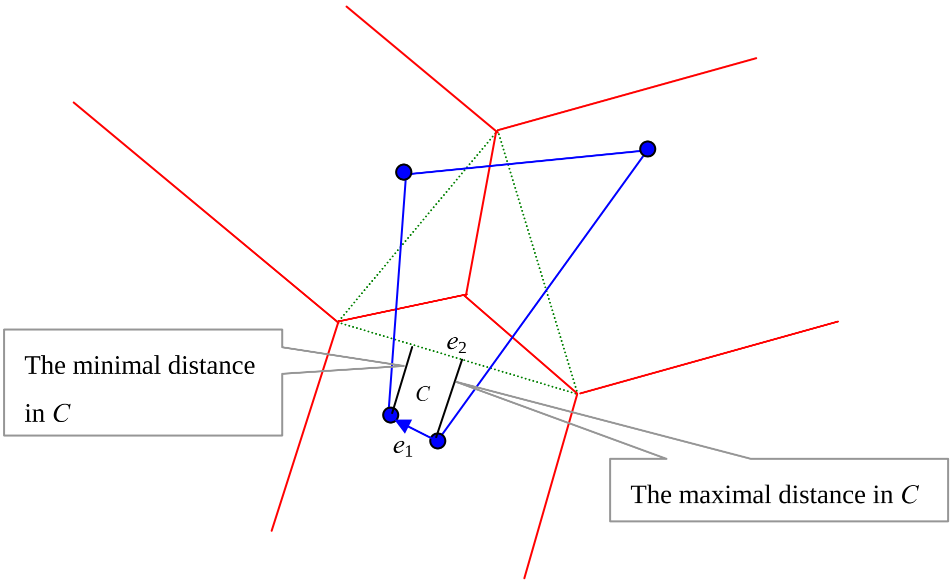


Fig. 2-5: The Voronoi-cell C and the possible positions for $\tilde{\delta}_H(I(Q), P)$

From Fig. 5, we will easily see that the distance between the corresponding element e_2 of P defining C to e_1 is a bitonic function (i.e. first decreases and then increases monotonically or just monotone decreases or just monotone increases, there is an instance of bitonic function at Fig. 6). It follows that the maximal distance of a point of Q on this edge e_1 to P must be at the endpoint of this edge e_1 or at the intersection point with some Voronoi-edge bounding cell C. (The property of a bitonic function) Therefore, if we use this concept to all the Voronoi-cells of Vor(P), we'll get all the possible positions where $\tilde{\delta}_H(I(Q), P)$ occurs. Similarly, we can obtain all the possible positions where $\tilde{\delta}_H(P, I(Q))$ occurs. We focus on these points (by applying 2 plane sweeps from right to left and left to right) and determine their distances. Finally we'll get $\delta_H(P, I(Q))$. We shall note that we just concentrate on problem P_1 that only allows the isometry $I(x) = t$, where t is a fixed translation.

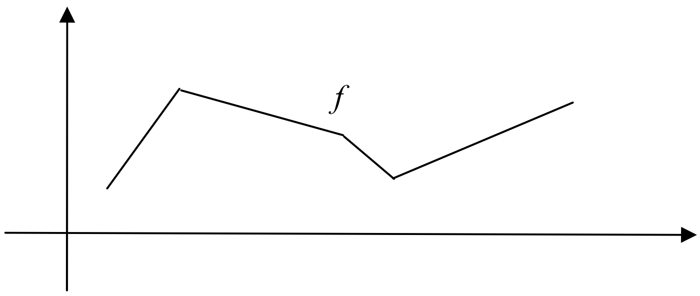


Fig. 2-6: An example of a bitonic function f which has only one intersection point with any vertical line

The instance in this paper is listed below:

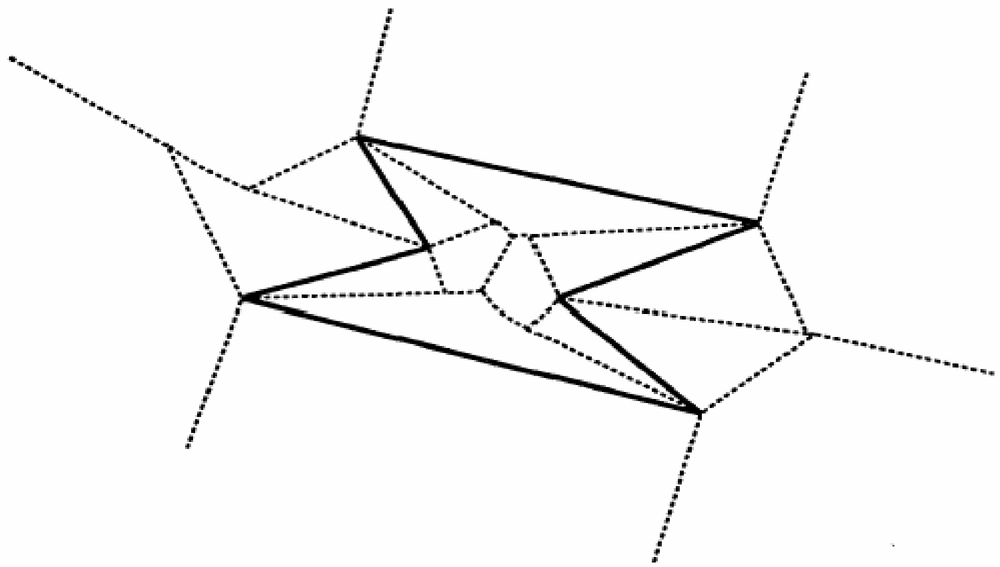


Fig. 2-7: A Voronoi diagram of a Polygon P

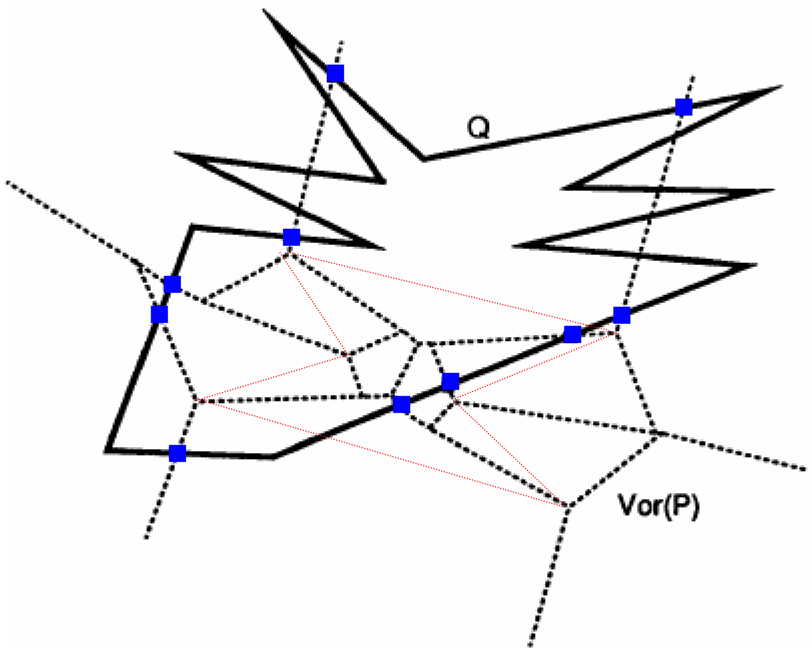


Fig. 2-8: Vor(*P*) and *Q*. ■ : the extreme intersection points ......: Original Polygon *P*.

Note that ■ is obtained as the following step:

In order to obtain only the extreme intersection points of each edge *e* of Vor(*P*), we delete e from the data structure as soon as the first intersection point with *Q* has been found.

(b) Suppose *P* and *Q* has *p* and *q* vertices respectively. The number of points that we'll consider to be the candidates of $\tilde{\delta}_H(Q,P)$ is $O(p+q)$ because (1) the number of vertices of *P* is *p*, (2) the number of vertices of *Q* is *q*, and (3) the number of some intersection point of *Q* with some Voronoi-edge *s* of *P* having either the smallest or largest *x*-coordinate among the intersection points of *Q* with *e* can be considered to be a constant. Therefore, we will obtain a $O((p+q)\log(p+q))$ algorithm **(because we have to determine the candidates having the smallest or largest *x*-coordinate. Furthermore, applying a sorting algorithm requires** $O((p+q)\log(p+q))$ **time)** which sweeps across the plane to determine all candidates.

Section Three – An Algorithm for P_1

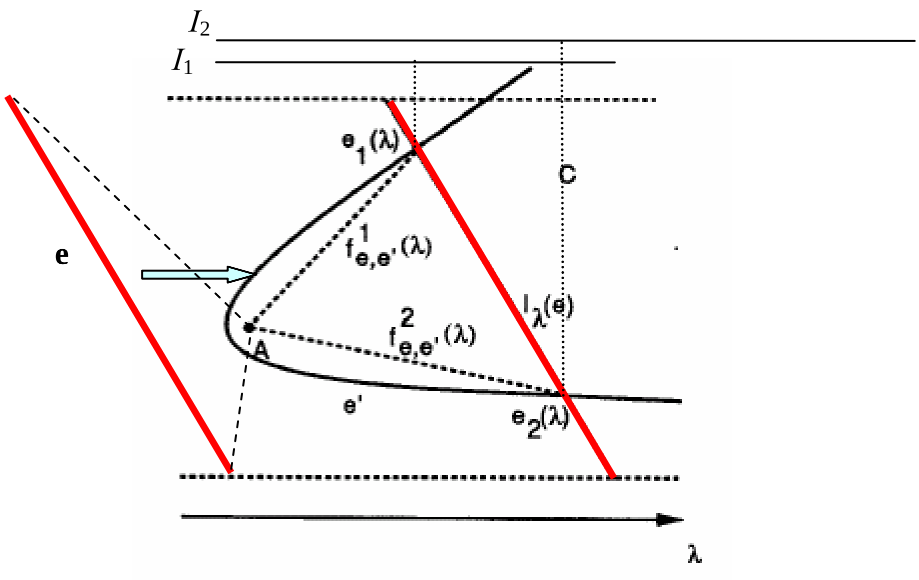


Fig. 3-1(1): A Voronoi-cell intersects an edge *e* of *Q* and *e* moves parallel to λ

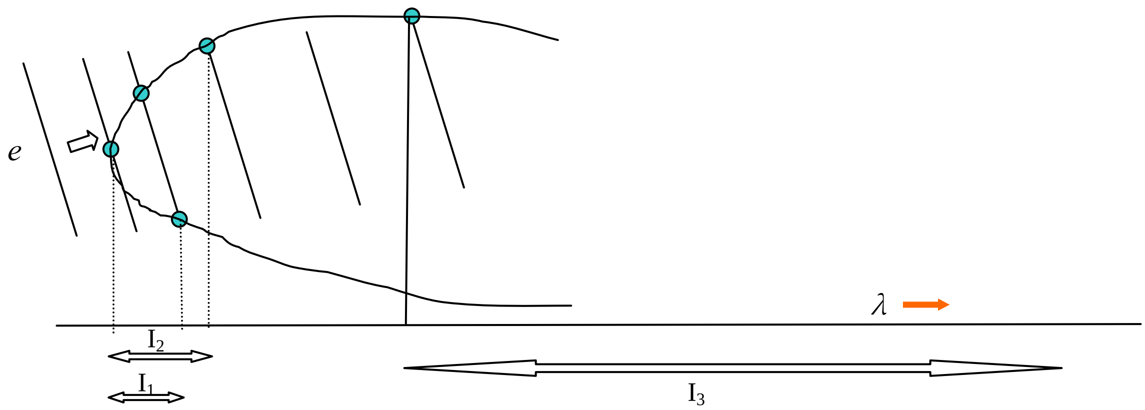


Fig. 3-1(2): An Idea proposed by Prof. Hwang

I. <u>Some definitions:</u>

(a) $I_\lambda(e)$ denotes the image of e moving parallel to a vector λ which is assumed to be (λ, 0), $\lambda \in \Re$. Somehow, λ can be viewed as a "*coordinate*",

(b) e is an edge of Q and e' is an edge of Vor(P) bounding cell C. There exist at most 3 intervals I_1, I_2 and I_3 (that may overlap an even coincide) such that for any $\lambda \in I_j$, $j=1,2,3$, there exists an intersection point between $I_\lambda(e)$ and e'. For $\lambda \in I_j$, $j=1,2,3$, denote by $e_j(\lambda)$ the corresponding intersection point.

(c) $f^{(j)}_{e,e'}(\lambda)$ is the function which assigns the distance of $e_j(\lambda)$ to the vertex or edge A of P defining cell C.

(d) Likewise, we define the function $f_{a,C}$ for each pair (a, C), where a is an endpoint of some edge e of Q and C a cell of Vor(P).
I.e. if the corresponding endpoint a is contained in C, $f_{a,C}(\lambda)$ is defined as the distance of this point to the site defining C.

From Lemma 1 we'll know that $\tilde{\delta}_H(I_\lambda(Q), P)$ is the maximum of all these functions above. It can be observed that the functions defined above are pieces of algebraic functions of degree at most 8. (Why is it at most 8? I asked Prof. Hwang this question and email to the author Alt, H.. Prof. Hwang thought that this is nontrivial and hard to compute the actual functions. Now I plan to omit this part first and wait for the reply of Prof. Alt.) Therefore any two of these functions intersect in at most 64 points. Consequently, by using the terminology from the theory of Davenport-Schinzel sequences, we can obtain their upper-envelope ***f*** consists of **at most** $\boldsymbol{\lambda_{66}(pq)}$ **pieces of the graphs** of the single functions and it can be computed in time $O(\lambda_{66}(pq)\log(pq))$. (*[A85]* ***Some Dynamic Computational Geometry Problems****, Atallah, M. J.,* ***Computers & Mathematics withApplications.****, Vol. 11, 1985, pp. 1171-1181.*) Likewise we can determine the distance from P to $I_\lambda(Q)$ as a function ***g*** of λ. No explicit expression for this function is known, but this function is bounded by $O(pq\log(pq)\log^*(pq))$. ([ASS89])

This part can be seen at:

> *[ASS89]* ***Sharp Upper bound and Lower Bound on the Length of General Davenport-Schinzel Sequences****, Agarwal, P. K., Sharir, M. and Shor, P.,* ***Journal of Combinatorial Theory Series A****, Vol. 52, 1989, pp. 228-274.*

and

> *[AS95]* ***Davenport-Schinzel Sequences and Their Geometric Applications****, Agarwal, P. K. and Sharir, M.,* ***Department of Computer Science, Duke University, Durham, North Carolina****, 27708-0129, September 1, 1995.*

The rest part of Section Three is going to introduce the concepts of Davenport-Schinzel sequences which are used in this paper. We can omit it because this part comes from other papers and is hard to understand.

<u>Definition of Davenport-Schinzel sequences:</u>

A Davenport-Schinzel sequence (DS(n, s) sequence) $U = (u_1, \ldots, u_m)$ is a sequence composed of n distinct symbols which satisfies the following two conditions:

(1) $\forall i < m,\ u_i \neq u_{i+1}$.

(2) There do not exist s + 2 indices $1 \le i_1 < i_2 \cdots < i_{s+2} \le m$ such that

$$\begin{matrix} u_{i_1} = u_{i_3} = u_{i_5} = \cdots = a, \\ u_{i_2} = u_{i_4} = u_{i_6} = \cdots = b \end{matrix} \quad \text{and} \quad a \neq b$$

(There should be no alternating subsequence of the form …a…b…a…b…of length s + 2 for any pair of characters a, b.)

For example, a instance of DS(n, 1) is $a(n)$ = 1, 2, 3, 4, 5, ⋯ , n. But $b(n)$ = 1, 2, 3, 1, 5, 6, ⋯ , n is not an instance of DS(n, 1) because b(1) = b(4) =1 and b(2) = 2≠1

We define $\lambda_s(n) = \max\{|U|:\ U \text{ is a DS}(n, s)\text{-sequence}\}$.

Let $F = \{f_1, \ldots, f_n\}$ be a collection of <u>partially defined</u> and continuous functions, so that the domain of definition of each function f_i is an interval I_i. Suppose each pair of these functions intersect in at most s points. We define ***the lower-envelope*** of F as

$$E_F(x) = \min f_i(x),$$

where the minimum is taken over those functions that are defined at x. For example, let's go to see Fig. 3-2.

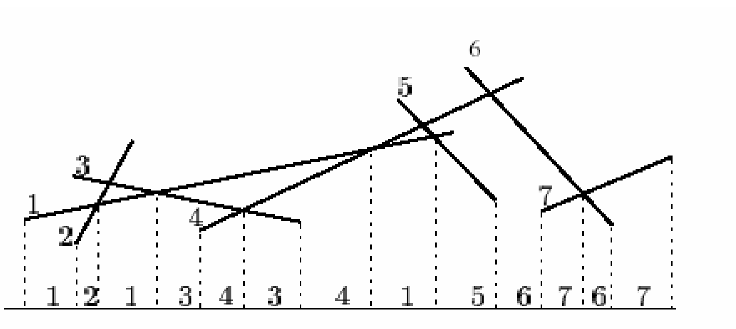


Fig. 3-2: The lower-envelope sequence

LES($\{f_1, f_2, f_3, f_4\}$) = <4, 2, 4, 1, 3, 2>. Similarly, we can obtain the upper-envelope **UE**$_F$ and the upper-envelope sequence **UES**(F) of F in a fully symmetric manner. Now, let's go to see the following theorem:

<u>**Theorem:**</u>

Let F be a collection of n partially defined, continuous, univariate functions, with at most s intersection points between the graph of any pair. Then $|\mathbf{UES}(F)| \le \lambda_{s+2}(n)$

Therefore we can easily figure out how and why $\lambda_{66}(pq)$ appears. (∵ Each two functions defined above intersect at most 64 points.) Yet how the degree 8 is produced is still puzzling.

Section Four – Algorithms for P_2 and P_3

I. This paper only briefly describes algorithms for P_2 and P_3.

P_2: $I = t$, where t is a translation.

P_3: The general problem: $I(x) = M \cdot x + t$

II. Suppose that two polygons P and Q are given as an instance for the problem P_2 and that I is an isometry (translation) such that $\delta := \delta_H(P, I(Q))$ is minimal. Let a, b be points of P and $I(Q)$ respectively, between which δ occurs. We call (a, b) a critical pair and the vector between a and b a critical vector. For example, let's see Fig. , let's see Fig. 4-1:

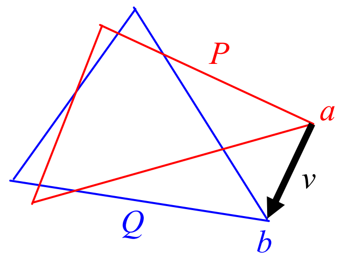


Fig. 4-1: (a, b) is a critical pair and v is a critical vector

And then we have a lemma that:

<u>Lemma2:</u>

Either there are two critical pairs such that for the corresponding critical vectors u, v, $u = -v$, or there are at least three critical pairs.

The proof of Lemma 2 can be done by contradiction considering the cases, where there are only 2 critical pairs. The proof is omitted in this paper, but we can have a feeling by the following figure Fig. 4-2.

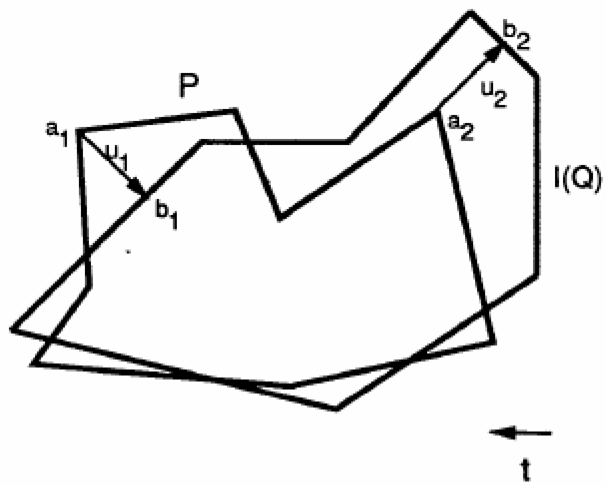


Fig. 4-2: Nonoptimal placement of Q, a translation by vector t will decrease the Hausdorff distance

If we move $I(Q)$ through the direction of vector t, we'll find that $d(u_1, b_1)$ and $d(u_2, a_2)$ decrease simultaneously. In other words, $\delta_H(P,Q)$ can be decreased, so it is not optimal.

If there are two vectors u, v, where $u = -v$, we have six cases about u and v:

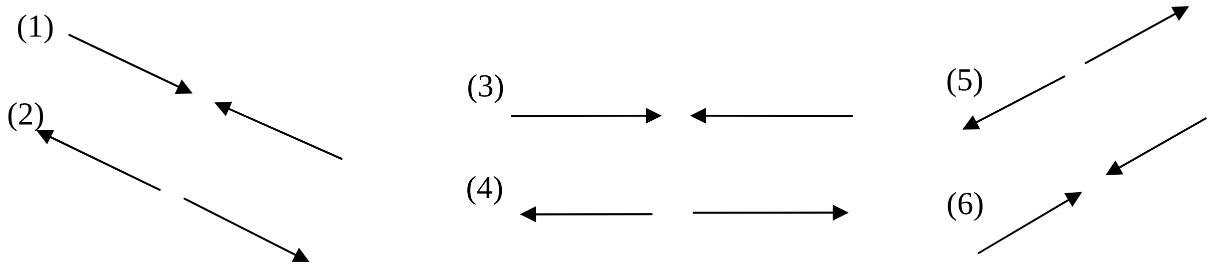


Fig. 4-3: Six cases of u and v

If u and v are in the cases above, we can easily find a translation that makes u shorter and v longer, or u longer and v shorter. For example, let's see Fig. 4-4.

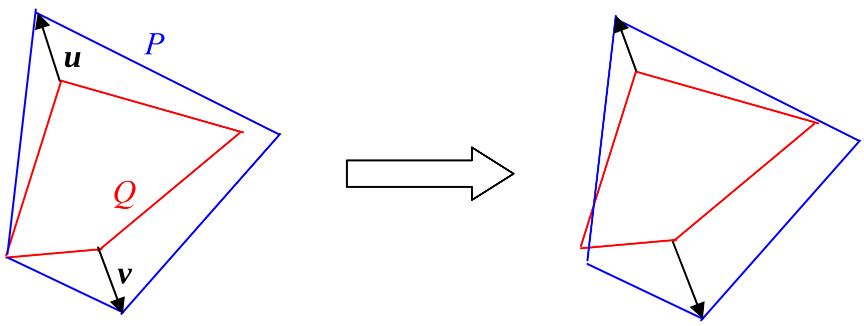


Fig. 4-4: One case for $u = -v$, where u becomes shorter and v becomes longer

The concept of the algorithm solving P2 is as follows:

(1) Consider all possibilities of elements of P, $I(Q)$ containing three critical pairs. We will only describe the case here where critical pairs (a_1, b_1), (s_2, b_2), (a_3, b_3) involve three vertices a_1, a_2 and a_3 of P, because the other cases can be solve with similar techniques within the same time.

(2) Therefore, consider all triples (a_1, r_1), (a_2, r_2), (a_3, r_3), where a_1, a_2, a_3 are vertices of P or $I(Q)$ and r_1, r_2, r_3 are vertices or edges of the other polygon. Because a_i has p or q possibilities and r_i has $2q$ or $2p$ possibilities respectively, there are $O((pq)^3)$ such triples. (Each critical pair has $O(2pq)$ possibilities, so the triple has $O(2^3(pq)(pq)(pq)) = O((pq)^3)$ possibilities.) If we demand that the three distances between a_i and r_i, i = 1, 2, 3, are the same, we obtain two equations

$$\begin{aligned} d(a_1,r_1) &= d(a_2,r_2) \\ d(a_2,r_2) &= d(a_3,r_3) \end{aligned},$$

Of course, d is the Euclidean distance if r_i is a point, or a shortest distance if r_i is a line segment.

In another point of view, since in problem P2 only two parameters specify the isometry, (This is because that I = t, where $t = \begin{bmatrix} t_x \\ t_y \end{bmatrix}$, $t_x, t_y \in \Re$, is an translation with two parameters t_x and t_y.) we need two equations, i.e., three critical pairs, to solve I.

Since in problem P_2 only two parameters specify I, there will be constantly many isometries such that the two equations above are satisfied. Finally for each of them, we determine $\delta_H(P, I(Q))$ by applying the algorithm mentioned in Section Two. So the algorithm has a run time of $O((pq)^3(p+q)\log(p+q))$. For problem P_3 we need four critical pairs, so P_3 can be solved by three equations similarly by applying the algorithm mentioned in Section Two in time $O((pq)^4(p+q)\log(p+q))$. (Algorithm in section 2: )

Section Five – Pseudo-optimal solutions:

I. Some definitions & lemmas:

Definition:

An algorithm is said to produce a ***pseudo-optimal solution*** for problems P_2 (P_3), iff there is a constant c > 0 such that on input P, Q the algorithm finds a translation (isometry) I with $\delta_H(P, I(Q)) \le c \cdot \delta$, where δ is the minimal Hausdorff-distance determined by the optimal solution.

Definition:

We claim that $\alpha(t)$ is a ***natural parameterization*** of a convex hull $\tilde{P}$ if $\alpha : [0, L_{\tilde{P}}] \to \Re^2$, where $L_{\tilde{P}}$ is the length of its edges, for any $t \in [0, L_{\tilde{P}}]$, the arc-length from point $\alpha(0)$ on $\tilde{P}$ to $\alpha(t)$ on $\tilde{P}$ equals t, $\alpha(0) = \alpha(L_{\tilde{P}})$ and the image of α equals $\tilde{P}$.

By the above definition, we can define the centroid S_P of the edges of the convex hull $\tilde{P}$ as follows:

$$S_P = \frac{1}{L_{\tilde{P}}} \int_0^{L_{\tilde{P}}} \alpha(t)dt,$$

where α is a natural parameterization of $\tilde{P}$.

Definition:

Let C_1, C_2 be curves. Then we define the ***continuous distance*** (This concept comes from the "*Fréchet Distance*"):

$$\delta_C(C_1, C_2) := \inf_{\alpha}(\max\{d(n_{C_1}(t), \alpha(t)) \mid t \in [0, L_{C_1}]\}),$$

where α ranges over all possible parameterizations, $\alpha : [0, L_{C_1}] \to \Re^2$, of C_2 and n_{C_1} is a natural parameterization of C_1.

δ_C can be imagined as follows:

Suppose there is a man walking his dog, the man walking on curve C_1, the dog on C_2. $\delta_C(C_1, C_2)$ is the minimal length of a leash that is possible.

II. Method 1: (A pseudo-optimal solution for problem P_2)

Let $r_p := (x_p, y_p)$ where $x_p(y_p)$ is the smallest x-coordinate (y-coordinate) of all points in P.

Let's see Fig. 5-1.

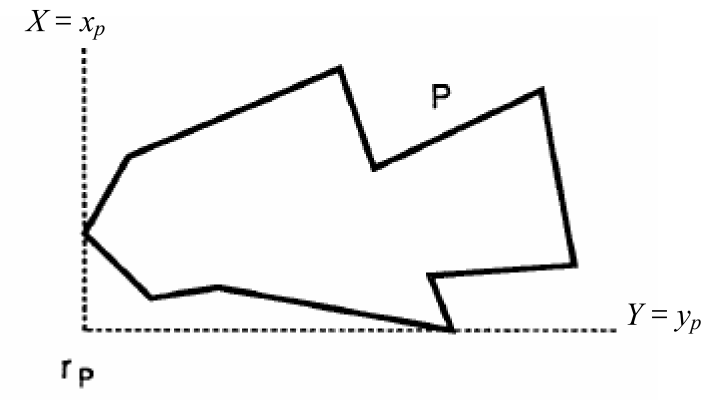
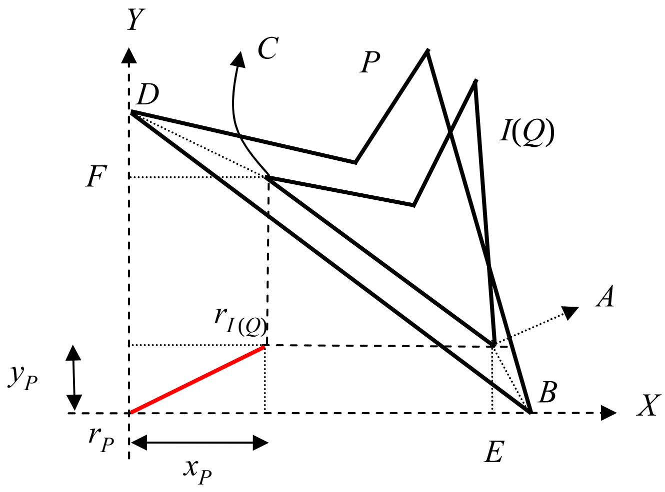


Fig. 5-1: $r_p = (x_p, y_p)$ of a polygon P

Suppose that P and Q are two polygons and I is a solution to P_2. Let $\delta := \delta_H(P, I(Q))$. Since $d(r_P, rI(Q)) \le \sqrt{2} \cdot \delta$. (See Fig. 5-2.) Therefore if $\tilde{I}$ is a translation mapping r_Q onto r_P, we'll get $\delta_H(P, \tilde{I}(Q)) \le (1 + \sqrt{2})\delta$, i.e. $\tilde{I}$ is a pseudo-optimal solution, We can apply the algorithm for P_0 (By using the concepts of intersection of a Voronoi diagram and a polygon), then we will obtain a runtime of $O((p+q)\log(p+q))$.



$$d(r_P, rI(Q)) = \sqrt{{x_P}^2 + {y_P}^2} = \sqrt{\overline{CF}^2 + \overline{AE}^2}$$
$$\leq \sqrt{\overline{CD}^2 + \overline{AB}^2} \leq \sqrt{\delta^2 + \delta^2} = \sqrt{2}\delta \cdot$$

Fig. 5-2: Why "$\sqrt{2}\delta$" occurs. (This proof was thought up as I sat in J. P. Lu's car.)

III. Method 2: (A pseudo-optimal solution for problem P_3)

We first define $\tilde{P}$ to be the convex hull of the polygon *P*. Next, we define S_P to be the centroid of the centroid the edges of the convex hull $\tilde{P}$, where S_P can be calculated as:

$$S_P = \frac{1}{L_{\tilde{P}}} \int_0^{L_{\tilde{P}}} \alpha(t)dt \, ,$$

where $\alpha : [0, L_{\tilde{P}}] \to \Re^2$ is a natural parameterization of $\tilde{P}$ such that the length from point α

(0) to α (*t*) equals *t*, and $L_{\tilde{P}}$ is the length of $\tilde{P}$.

Why? $\Longrightarrow$ ***A centroid of a polygon never changes under rotations.***

Suppose *Q* is another polygon and *I* is the isometry minimizing $\delta := \delta_H(P, I(Q))$. Since from the following lemma:

<u>Lemma 4:</u>

$$d(S_P, S_{I(Q)}) \le (4\pi + 3)\delta$$

If an isometry $\tilde{I}$ gives a minimal $\delta_H(P, \tilde{I}(Q))$ among the ones mapping S_Q onto S_P, we can obtain that $\delta_H(P, \tilde{I}(Q)) \le \delta + (4\pi + 3)\delta = ((4\pi + 4)\delta)$. The angle of rotation, which gives the optimal solution $\tilde{I}$, can be determined by a technique analogous to the one used for solving problem P_1. The running time is bounded by $O(\lambda_{66}(pq)\log(pq))$.

IV. Method 3: (A further better idea)

Now we are going to briefly introduce this improved idea. The above constant $(4\pi + 4)\delta$ can be reduced to any fixed constant $c > 1$ without increasing the asymptotic runtime as follows:

By lemma 4, we know that the optimal isometry I maps S_Q into the $(4\pi + 4)\delta$ - neighborhood U of S_P. We place onto U a **sufficiently small grid** so that no point in U has distance greater than $(c - 1)\delta$ from a gridpoint. (See Fig. 5-2) There are constantly many gridpoints within U.. Then, we place S_Q instead of onto S_P only, onto ***each one of these gridpoints*** and proceed as described before. It follows from previous discussion that for the solution $\tilde{I}$ found this way holds:

$$\delta_H(P, \tilde{I}(Q)) \le c\delta.$$

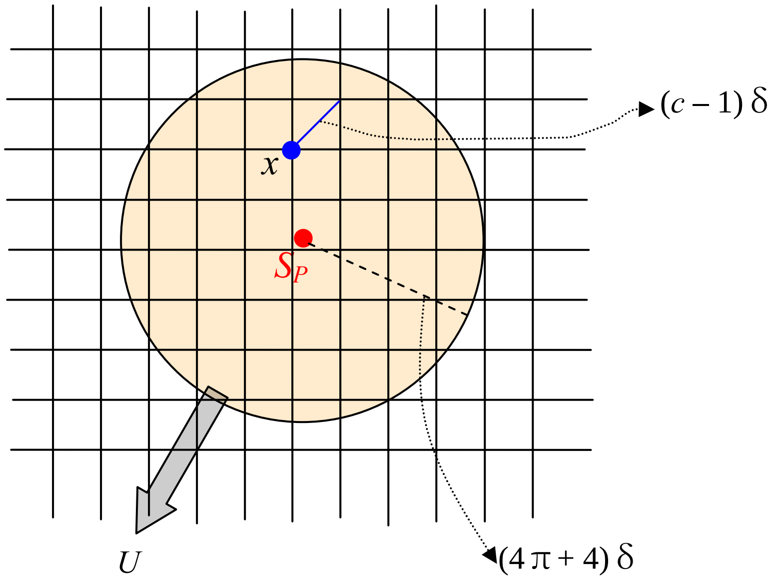


Fig. 5-3: Grid, gridpoints, S_P and the region of the $(4\pi + 4)\delta$ - neighborhood U of S_P