Optimal Algorithms for Finding the Symmetries of a Planar Point Set

P. T. Highnam

CMU-RI-TR-85-13

Department of Computer Science
The Robotics Institute
Carnegie-Mellon University
Pittsburgh, Pennsylvania 15213

August 1985

Table of Contents

Abstract

We present an asymptotically optimal algorithm to locate all the axes of mirror symmetry of a planar p
t. The algorithm was derived by reducing the 2-D symmetry problem to linear pattern-matching. Opt
gorithms for finding rotational symmetries and deciding whether a point symmetry exists are also preser

Introduction

^ set of points P in the plane has an axis of mirror symmetry A when for every point *p* of P not lying 01 re is another point p^1 in P s.t. A is the perpendicular bisector of the line pp^f. P has a rotational symm< a when rotating P about its centroid C by *a* is an identity operation on P. P has a point symmetry (wh ist be at C) precisely when P has a rotational symmetry of *m* .

Tiis note presents an optimal O(n log n) algorithm for discovering all the mirror symmetries of an n p< P in detail and describes the changes needed to detect the rotational symmetries (and hence the existe a point symmetry). Lower bounds are shown for each problem. Mirror symmetries are the objects srest until section 5.

Tie 2-D mirror symmetry problem is reduced to a 1-D pattern-matching problem for which fast soluti well-known. Any A must pass through the centroid C of P, so the points are first translated so that (I responds to C. After expressing the points in polar coordinates, sort them increasing-distance-witf reasing angle from a reference direction. For each unique angle (at most n), replace the set of po iding at that angle (i.e., $\geq$ 1 point) by a *tuple* which is simply a list of their distance components. Let Tiber of unique angles be m ($\leq$ n). Consider the result as a length 2m string F, the symbols of which srnately tuples and the angles between adjacent tuples. The mirror symmetries of P correspond exactl] length 2m subsequences of FF which are palindromes.

Tie palindromes can be discovered by a fast one-dimensional string matching algorithm, looking urrences of the reversal of F in FF. One such algorithm is that of Knuth, Morris and Pratt [1] ("KM ich permits the detection of all occurrences of a *pattern* within a *text* in time proportional to the sum of gths of the text and pattern.

lection 2 of this report contains the algorithm, and section 3 contains proof of why the algorithm works, tion 4 we demonstrate that it is impossible to improve the asymptotic time bound. In section 5 we pres ariant of the basic algorithm which detects the rotational symmetries and point symmetry, and a proof I se are also optimal.

2. Algorithm

Input : n point planar point set P, (p_i for i =0(1)n-1).

Output : The centroid of P and the orientations of each of its axes of mirror symmetry.

1. Find the centroid C of P. Translate the point set so that the origin coincides with C.

2. Select an arbitrary reference direction (for convenience we choose the direction of p_0). F the points in polar coordinates with the angle component as measured anticlockwise reference direction, denote point p_i as (r_i,θ_i).

3. Sort the points by increasing-distance-within-increasing-angle. Delete any points that l distance (r_i = 0). Let the number of different angles be $m \leq n$.

4. For each of the m different angles, represent its set of points in a single *tuple* , which sim the set of distances at that angle. Beginning at the reference direction, proceed through tuples in order of increasing angle, generating the length 2m string F : at the current append the tuple to the string; move to the next tuple, appending the angle traversed to t Finish when returning to the reference direction. The first element of this string is a tup and tuples alternate within it. Create the length 2m string R and the length 4m-2 stri shown below:

$$F = f_0 f_1 \dots f_{2m-1}$$

$$R = f_0 f_{2m-1} f_{2m-2} \dots f_1$$

$$F' = f_0 f_1 \dots f_{2m-1} f_0 f_1 \dots f_{2m-3}$$

5. Employ a string-matching algorithm such as KMP to locate all of the matches of R in F the definition of F the only matches are possible beginning at even indices (since f_0 is Denote the size t ($0 \leq t \leq m$) list of indices of F$'$ at which a match can begin by I_j (j =0(

6. Compute for each match index I_j the orientation of an axis of mirror symmetry A_j. Let k

k_j is even => A_j passes through tuple f_{k_j} .

k_j is odd => A_j bisects angle f_{k_j} .

3. Details

Because the tuples do not necessarily have a unique distance associated with them, we can consider the tuples $\{f_{2i}$, for $i = 0$ to $m\text{-}1\}$ as laid out on the perimeter of a circle (diameter unimportant, centre at ntroid of the point set), with tuples f_{2i} and f_{2i+2} separated by the angle f_{2i+1}. Tuples f_{2m-2} and f_0 parated by angle f_{2m-1}. The axes of this representation are precisely the axes of the original point set.

The notation used is that of the algorithm of section 2. $F = f_0 f_1 \ldots f_{2m-1}$, where m is the number of tup d an element f_i is either an angle (i is odd), or a tuple (i is even). $R = f_0 f_{2m-1} f_{2m-2} \ldots f_1$, a $= f_0 f_1 \ldots f_{2m-1} f_0 f_1 \ldots f_{2m-3}$. We say that R matches F' at index i when the length 2m substring of . $f_{2m-1} f_0 .. f_{i-1}$ matches R. For all such matches i must be even because f_0 is a tuple and the tuples pear with even index in F.

MMA For any k $(0 \leq k < m)$: F is a k-palindrome *iff* R matches F' at index $2k$.

Define substrings,

$$a_1 = f_{k+1} \ldots f_{2k-1} \qquad \text{(Length } k\text{-1)}$$

$$a_2 = f_{2k+1} \ldots f_{2m-1} \qquad \text{(Length 2m-2}k\text{-1)}$$

$$a_3 = f_0 \ldots f_{k-1} \qquad \text{(Length } k\text{)}$$

$$b_1 = f_{k-1} \ldots f_1 \qquad \text{(Length } k\text{-1)}$$

$$b_2 = f_{2m-1} \ldots f_{2k+1} \qquad \text{(Length 2m-2}k\text{-1)}$$

$$b_3 = f_{2k} \ldots f_{k+1} \qquad \text{(Length } k\text{)}$$

When F is a k-palindrome and $k < m$ we know that $f_{k+1} .. f_{k-1}$ is a palindrome. More specifica $f_{2k} a_2 a_3 = b_1 f_0 b_2 b_3$. Therefore $f_{2k} = f_0$, $a_1 = b_1$, $a_2 = b_2$, and $a_3 = b_3$ using the substring leng ven above.

When R matches F' at index $2k$ we know f a a f a = f b b f b. The equivalences here are ag

LEMMA For any k $(0 \leq k < m)$: F is a k-palindrome *iff* k is even and there is an through tuple f_k, or k is odd and there is an axis A bisecting the angle f_k.

There are two cases to consider,

- k is odd. Let $k = 2i+1$. That F is a $(2i+1)$-palindrome is precisely the condition broug by A bisecting angle f_{2i+1}.
- k is even. Let $k = 2i$. That F is a $2i$-palindrome is precisely the condition brought ab passing through tuple f_{2i}.

We have shown by the two lemmas that for any k $(0 \leq k < m)$, there is an axis A passing thro is even), or an axis A bisecting angle f_k (k is odd) precisely when there is a match for R in F' a remains to point out that this is sufficient to capture all axes because an axis A must cut the circl π apart and in having k range from 0 to m-1 we have covered a complete semi-circle.

4. Complexity

In this section we show that the complexity of the algorithm presented in Section 2 match problem and so is optimal, i.e., the algorithm is O(n log n) and the problem is shown to be Ω show the latter we use a reduction of the set equivalence problem to that of deciding whether th an axis of mirror symmetry in a planar point set. This also shows that it is no harder to find mirror symmetry than it is to find whether there are any at all.

From the description of the algorithm in section 2 we can see that it is O(n log n), and this de sorting operation of *Step 3*. *Steps 1,2,4* and *6* are clearly O(n) while *Step 5* can be done in O(n string matching algorithm such as KMP. The worst-case complexity of the KMP algorithm is n alphabet size.

from B using L_B.

3. Manufacture the 2D point set P defined as $\{(a, a) : a \in A'\} \cup \{(b, -b) : b \in B'\}$.

4. Find the axes of mirror symmetry of the set P. If P has an axis of mirror symmetry then A and B are the same, else they are not.

The set P defined above will have at most one axis of mirror symmetry and this if present will correspon[d] [to th]e 'x-axis' of the 2D system created.

Rotational and point symmetries

[I]n this section we present a version of the algorithm of section 2 which will find all the rotati[onal sym]metries of point set P. Hence, as noted in the introduction, we will also have discovered whether the[re i]s a point-symmetry because that is identical to a rotational symmetry of π.

[A] rotational symmetry is an angle α s.t. rotation of P by α about its centroid is an identity operation o[n P. Fu]rthermore, if α_0 is the smallest such angle then the set of angles of rotational symmetry is precisely the integer multiples of α_0. Hence, it is only necessary to find α_0.

[G]iven the string representation F of P, proceed by matching F against F', starting with the first elemen[t] [a]t f_2 and moving to the right. This avoids a meaningless match of F with itself as prefix of F' and beca[use the an]gle α_0 corresponds to the lowest index at which F will match F'. Stop when the first match is found ([or] [fai]lure occurred at the end of F', implying no rotations). Let the index of the match in F' be i, which m[ust be] even as before. The corresponding α_0 is the sum of the angles in F' lying to the left of the ma[tch,] [α_0] $= f_1 + f_3 + \ldots f_{i-1}$.

The complexity of this algorithm is the same as that of the mirror symmetry algorithm, O(n log [n). Ch]ecking whether α_0 is an integer divisor of π is constant time work and so O(n log n) applies to the po[int-sy]mmetry decision too.

We now show that deciding whether P is point symmetric is Ω(n log n) and so the algorithm above

3. Manufacture the 2-D point set P defined as $\{(a, a) : a \in A'\} \cup \{(-b, -b) : b \in B'\}$.

4. Decide whether P is point-symmetric, if it is then A and B are the same, else they are not.

6. Acknowledgments

Dan Hoey provided helpful discussion at the right time. It's been a very long time since this was and several people have made comments which have improved the presentation particularly G.J. Agin.

References

[1] D.E.Knuth,James H.Morris Jr. and V.R.Pratt, Fast Pattern Matching in Strings, *SIAM J COM.* 2 (June 1977), 323-350.