COMPUTER VISION, GRAPHICS, AND IMAGE PROCESSING **30**, 362–363 (1985)

NOTE

On the Time Complexity for Circumscribing a Convex Polygon

TONY C. WOO AND HYUN-CHAN LEE

Department of Industrial & Operations Engineering, University of Michigan, Ann Arbor, Michigan 48109

Received November 22, 1984; accepted November 28, 1984

A recent article "Circumscribing a Convex Polygon by a Polygon of Fewer Sides with Minimal Area Addition" by Dori and Ben-Bassat, *Comput. Vision Graph. Image Process.* **24**, 1983, 131–159, raised several interesting questions including the time complexity of their algorithm. In this paper, the time complexity on circumscribing an n-gon by an m-gon, where $m < n$, is analyzed to be $O(n \lg n)$.

This is a note on the time complexity analysis for circumscribing an n-sided convex polygon by an m-sided polygon with minimum area addition, where $m < n$. The interesting circumscribing algorithm was described by Dori and Ben-Bassat in "Circumscribing a Convex Polygon by a Polygon of Fewer Sides with Minimal Area Addition" (*Comput. Vision, Graph. Image Process.* **24**, 1983, 131–159). Rather than $O(n)$ as stated by the authors, we believe that the overall time complexity should be $O(n \lg n)$.

According to Algorithm 0, p. 144, two major steps take place. They are: single side reduction (Algorithm 1) and compression (Algorithm 3). As they both involve constrained single side reduction (Algorithm 2), it is useful to list the number of times Algorithm 2 is called by others.

Algo	Name	Calls	No. of times
0		1	$(n - m)$
		3	3
1	SSR	2	n or 4
2	CSSR	NIL	NIL
3	Compression	2	$p = O(n)$

Algorithm 1 calls Algorithm 2 either n times or at most four times depending on when Algorithm 0 calls Algorithm 1. In particular, the first time Algorithm 0 calls Algorithm 1, Algorithm 2 is carried out n times. During the subsequent $(n - m - 1)$ calls to Algorithm 1, Algorithm 2 is carried out at most 4 times. Given that the time complexity of Algorithm 2 is constant, it is understandable why one would arrive at the conclusion that Algorithm 0 is of linear time.

However, in step 2 of Algorithm 1, p. 141, the smallest $T_j^{(i)}$ is chosen. Though it only takes $(\lg n)$ time to find the minimum, the operation must be performed for

0734-189X/85 $3.00

every vertex. In particular, the worst case time complexity for step 2 of Algorithm 1 is

$$4 \sum_{i=1}^{n-m-1} \lg(n-i) \simeq 4(n-m-1)\lg(n-1).$$

The $(n \lg n)$ time complexity for step 2 of Algorithm 1 is reflected in the overall time complexity for Algorithm 0 in the following way. The first time Algorithm 1 is called, the time complexity is $(n + n\lg n) \simeq (n \lg n)$. The subsequent $(n-m-1)$ calls total $4(n-m-1) + 4(n-m-1)\lg(n-1) \simeq (n\lg n)$. Hence, the overall time complexity should be $O(n \lg n)$.

Algo	Name	Time Complexity
0		$O(n \lg n)$
1	SSR	$O(n \lg n)$
2	CSSR	Constant
3	Compression	$O(n)$