**MINI PROJECT**

*Submitted by*

**Divyansh Mohan**

**(RA2011051010059)**

**Praneet Mishra**

**(RA2011051010069)**

*Under the Guidance of*
**Dr.M.Ramprasath**

**ASSISTANT PROFESSOR**

**DEPARTMENT OF COMPUTING TECHNOLOGIES**

***In partial satisfaction of the requirements for the degree of***

**BACHELOR OF TECHNOLOGY**
**In**
**COMPUTER SCIENCE ENGINEERING**


**SCHOOL OF COMPUTING**

**COLLEGE OF ENGINEERING AND TECHNOLOGY**

**SRM INSTITUTE OF SCIENCE AND TECHNOLOGYKATTANKULATHUR – 603203**

**JUNE 2022**

SRM INSTITUTION OF SCIENCE AND TECHNOLOGY KATTANKULATHUR-603203

# BONAFIDE CERTIFICATE

Certified that this mini project titled **"Minimum area Polygonization"** is the bonafide work done by **Divyansh Mohan(RA2011051010059),Praneet Mishra(RA2011051010069)** who carried out the mini-project under my supervision. Certified further, that tothe best of my knowledge the work reported here in does not form part of any other work.

**SIGNATURE**

Dr.M.Ramprasath
**DAA – Course Faculty**
Assistant Professor
Department of Computing Technology

**HOD Signature**

# Minimum Area Polygonization

## DAA/18CSC204J - Mini Project


1. *Divyansh Mohan – RA2011051010059*
2. *Praneet Mishra – RA2011051010069*

# ***Abstract***

We consider an application of the divide and conquer technique to the MinimumArea Polygonalization (MAP) problem. It is known that MAP problem belongs to NP- Hard set of problems. In this paper we propose a heuristic algorithm for solving the Minimum Area Polygonalization problem that is based on recursive subdivision of the set of points into two smaller subsets, constructing approximated solutions for the subsets and merging them with respect to minimising the total area using the minimum area quadrilateral between two polygons. The algorithm of

finding such a quadrilateral is described at this paper as well. Time complexity of the entire algorithm is O ($n^2$) using O(n) memory.

# ***Problem statement***

The "Divide and Conquer" technique to Solve the Minimum Area Polygonalization Problem

# ***Existing system***

To date, a heuristics in combination with simple polygon generation algorithms, are used to solve the MAP problem .For example, forthe *Steady Growth algorithm*, the detailed

description of which can be found in, the following heuristic criteria are used:

1. Building an initial triangle:

(a) Random triangle

(b) Greedy triangle

2. Feasible point selection:

(a) Random feasible point

(b) Greedy feasible point

3. Edge selection: (a) Greedy edge

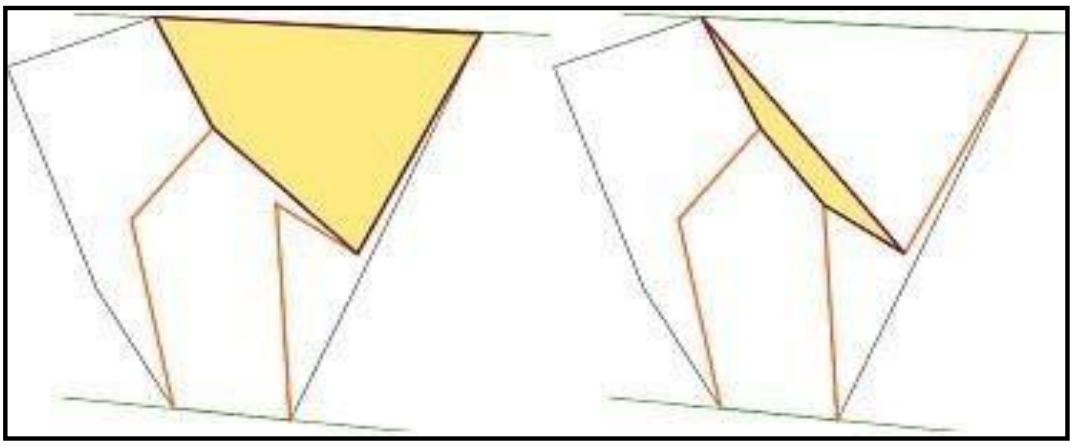
In this approach to the construction of an approximated solution is also described in detail. A modified version of this approach is proposed in, which has the $O(n^3)$ time complexity at the expanse of memory $O(n^2)$. Also, there is a randomized algorithm for solving theMAP problem with time complexity $O(n^2 \log n)$ using $O(n)$ memory .

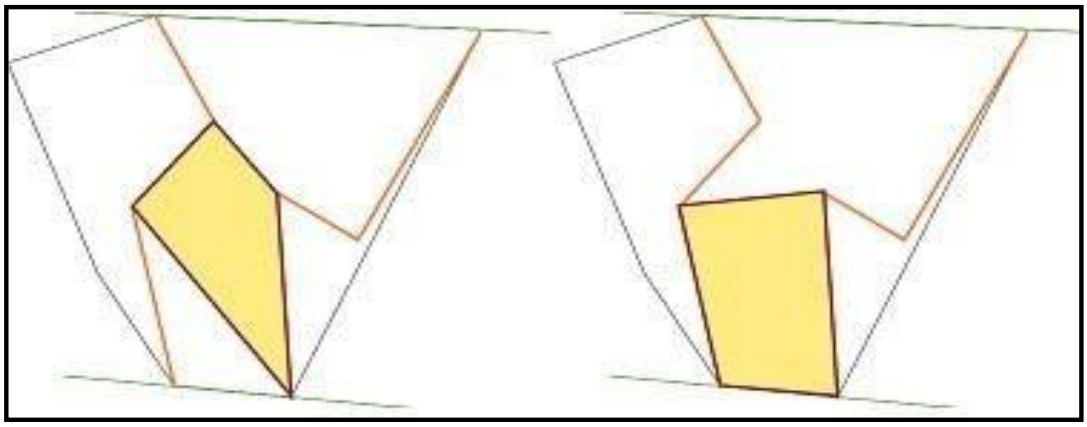
Authors formulate MAP problem as one of generalized problems of nonlinear programming models, and present a genetic algorithm to solve it. The only algorithm that gives an exact solution is "Permute and Reject". It generates all possible permutations of the set of points, for each permutation this algorithm checks whether it determines a simple polygon, and among of the generated simple polygons, the one that has the smallest area is chosen.

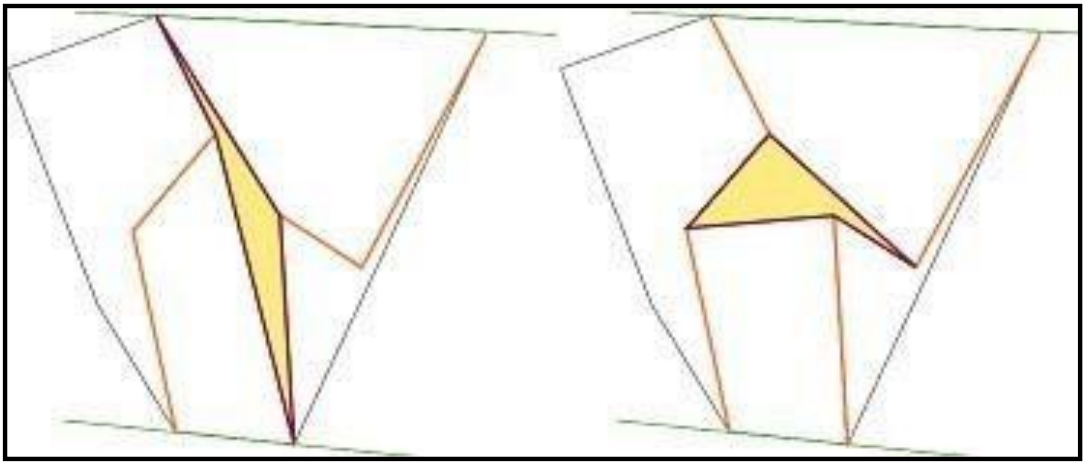
# ***Proposed System***

We propose an approach that allows to find an approximate solution for the MAP problem in polynomial time and to develop agreedy algorithm for approximating the solution of theMAP

problem using the "Divide and conquer" technique.







# ***Explanation***

Problem MAP: A set S of N points is given on a plane. It is necessary to construct a simple polygon with the smallest possible area, which would cover a given set of points, and which vertices are all points of the set S.

A. The algorithm idea:-

If number of elements in the set S is small enough, we can find an exact solution using any brute-force algorithm, like "Permute and Reject". Otherwise we divide the set S into two subsets S1 and S2, so that the number of elements in the subsets differs by no more than one. Find the solutions of the MAP problem for these two subsets recursively. Now we have approximations for sets S1 and S2, and we have to merge them into one

simple polygon, using a quadrilateral of the minimum area, so that the resulting polygon will not contain self- intersections.

Now we will describe necessary procedures used by the proposed algorithm for solving the MAPproblem.

## B. An algorithm for finding a minimal area quadrilateral between two non-intersectingsimple polygons:-

Without loss of generality, we assume that all vertices of the first polygon are to the leftof the vertices of the second one. Also, for definiteness, we assume that the vertices of polygons are given in the counterclockwise order. For those polygons construct the upper and lower tangent. Consider the sub-chain of the first polygon, which is from the point of intersection of the lower tangent to the upper, and the sub-chain of the second polygon, which is from the point of the intersection of the upper tangent to the lower one (Fig.1). Consider all possible quadrilaterals without self-

intersections, which contain one edge from each of sub-chains and do not cross the sub-chains in other points, and choose among of them the one with the smallest area.

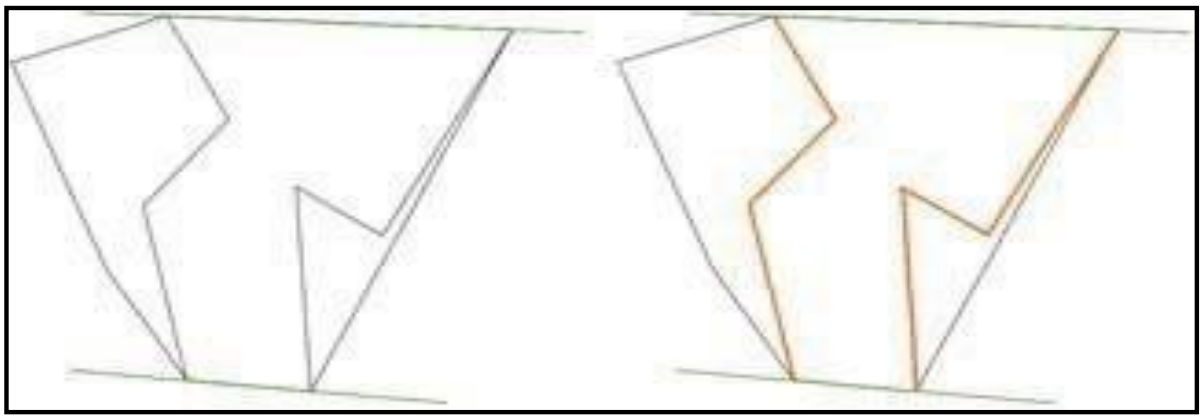


Fig1:- Two sub-chain between which a minimum area quadrilateral is sought.

Consider the polygon formed by the sub-chains, and segments of the upper and lower tangents. For each vertex of one of the sub-chains (for definiteness, we consider that the first one was chosen) construct a visibility polygon from this vertex (Fig. 2).

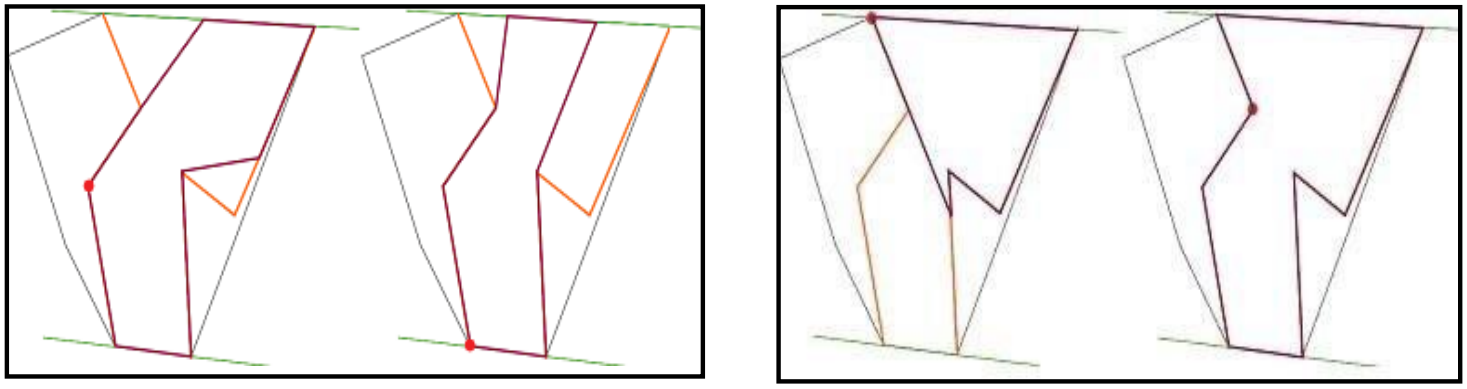
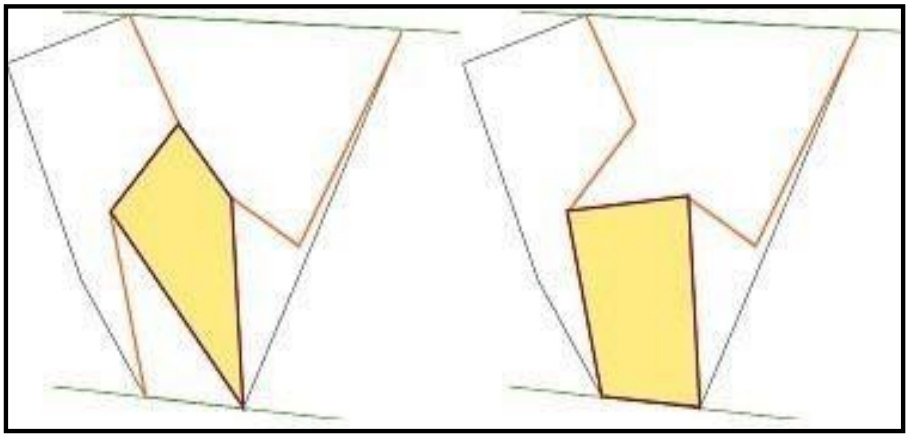


Fig.2.Visibility, polygons, for each vertex of the left sub-chain

Then, if for two edges (u1, u2) from the first sub-chain, and (v1, v2) from the second sub-chain, the vertex v1 is visible from the vertex u1, and the vertex v2 is visible from the vertex u2, and the segments (u1, v1), (u2, v2) do not intersect, then these edges define a correct quadrilateral (Fig.3) among of which we choose the having the smallest area (Fig. 4).



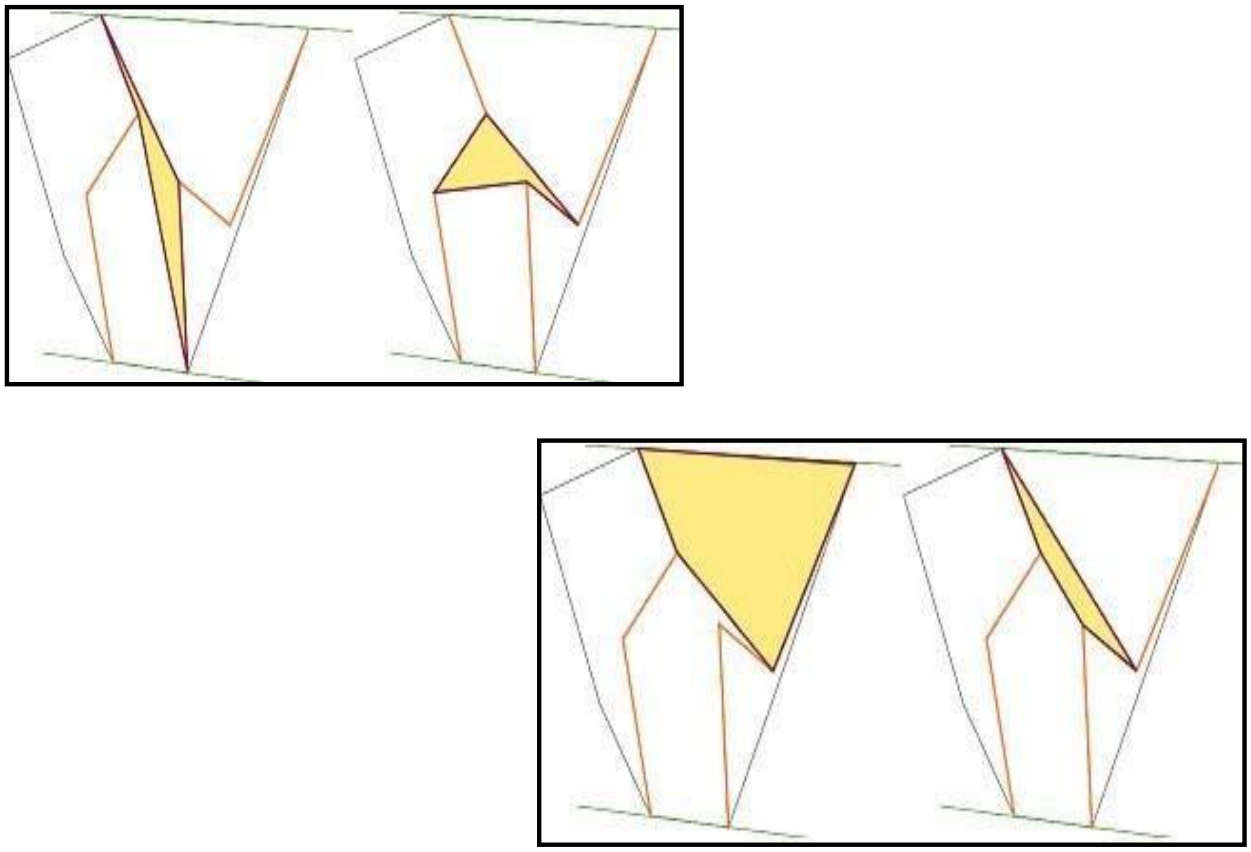


Fig3:- All possible quadrilateral

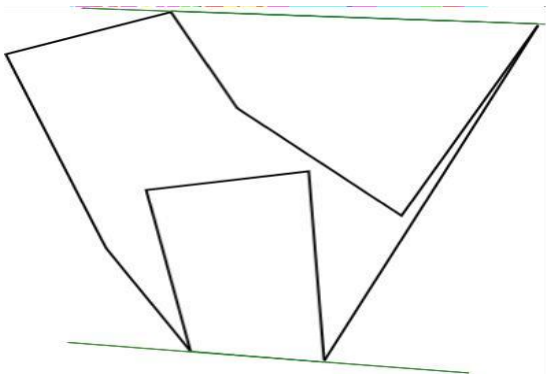


Fig4:- The resulting polygon

# ***Implementation***

*Algorithm:-*

Input: Set S of n points on a plane. Output: A simple polygon that approximates minimum area polygonalization of the given set S. Pre-processing: Sort out all points by the x-coordinate from left to right.

1. If |S» ≤ 5, then return the exact solution found by brute-force algorithm.

Otherwise, divide the set S into two parts S1 and S2. Find a solution for these subsets recursively.

2. Merge the solutions for the subsets S1 and S2 using Merge Simple Polygon procedure, this gives a solution for the entire set S. Procedure

Merge Simple Polygons: Input: Two simple polygons P11 and P2.

Output: A simple polygon, obtained by merging polygons P11 and P22.

1. Find convex hulls of polygons P1 and P2: CH(P11) and CH(P2).

2. Find upper and lower tangent for CH(P11) and CH(P22).

3. Find two sub-chains between two polygons: l11 and l22.

4. For each edge from l1: Find a set of visible vertices from the end of the segment. For each edge from l2: Check the Intersection With Sub-chains Test condition, and in the case of a successful passage, if area of the current

quadrilateral is smaller than all considered before it, then remember the current one.

5. Merge the polygons and remove two extra edges. Intersection With Sub-chains Test: Input: Edges (u11, u22) and (v11, v2). Output: True - if the edges define the correct quadrilateral, False - otherwise.

1. If segments (u11, v1) and (u2, v2) intersect - return False.

2. If vertex v1 is visible from vertex u1, and vertex v2 is visible from vertex u2, return True otherwise return False.

# ***Result analysis***

## *Time complexity:-*

The approximate solution of MAP problem can be found in O(n2) time with O(nlogn) pre-processing time.

Proof:- Pre-processing sorting out points of the set S takes O(nlogn) time . Since the algorithm uses the "Divide and Conquer" strategy, if the complexity of the merge function is O(f(n)), then the overall complexity of an algorithm is determined by the solution of the recurrence T(n) = 2T(n/2) + f(n). It makes sense to construct convex hull using the “Divide and Conquer” strategy as well. As we have two convex hulls from previous step, we can join them in O(n) time . During merging two convex polygons we also find two tangents between those polygons that are needed for Merge Simple Polygons procedure.

Construction of two sub-chains I1 and I2 requires traversing along the contour of the

corresponding polygons, therefore, it takes O(n) time. An algorithm that finds a quadrilateral of minimal area, which is used for merging two simple polygons, is described. It is possible to find the set of visible vertices of a polygon from a vertex in O(n) time, this procedure have to be performed for all vertices of one of the sub-chains, therefore the total complexity of finding the minimum area quadrilateral is O(n2). Thus, the time complexity of the merge procedure is O(n2 + n) = O(n2). That is, we have the recurrence T(n) = 2T(n/2) + n2. This recurrence can be solved using Master Theorem [18], and the solution is O(n2).

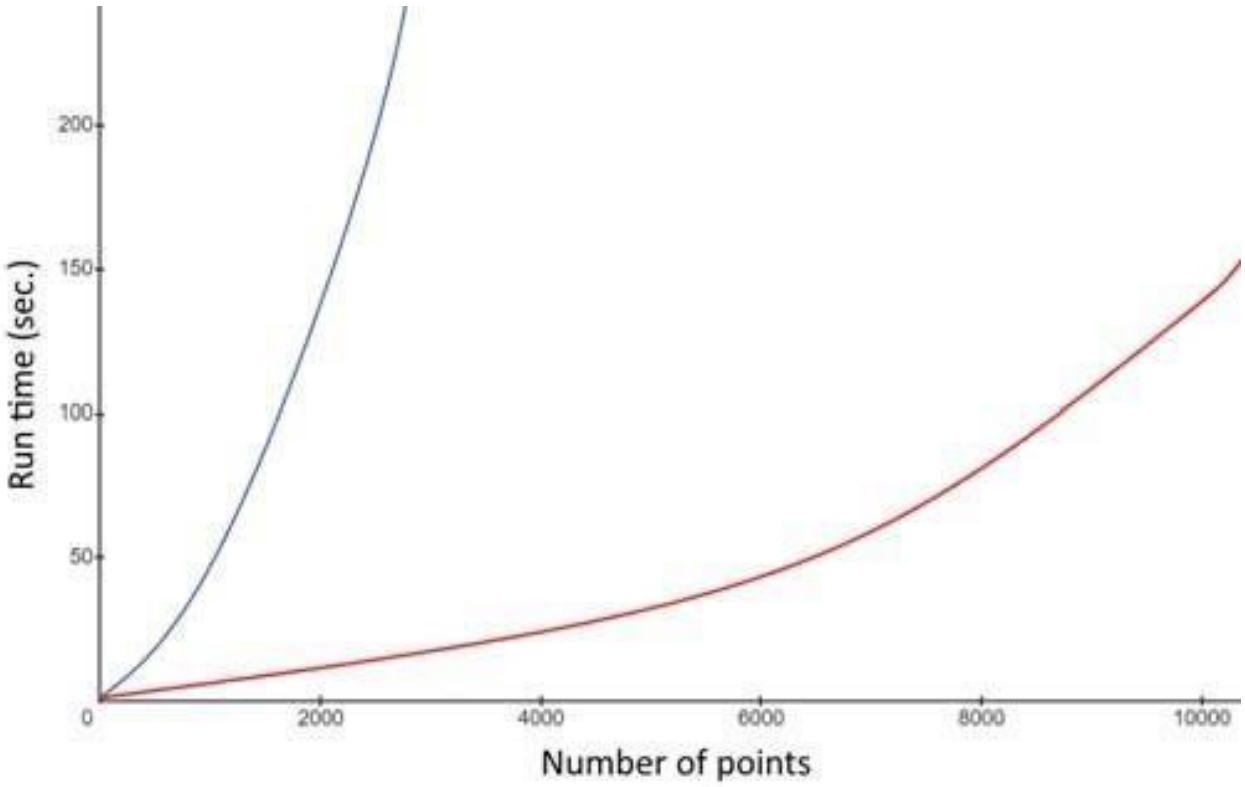



Fig. 8. Running time comparison: our approach (red curve) and “Greedy triangle” (blue curve).

# ***Conclusion***

In the paper, a heuristic algorithm for constructing an approximate solution of the Minimum Area Polygonalization problem is proposed. The algorithm is based on the Divide and Conquer strategy. The described algorithm can be used in various application areas where the MAP problem arises. For an instance minimum area can be used as one of the restrictions on polygonalization in pattern recognition and image reconstruction problems [19]. We have proved that the time complexity of the algorithm is O(n2), pre-processing phase complexity is O(nlogn) and memory usage O(n). This follows that the given algorithm is one of the fastest algorithms for constructing the approximate solution of the MAP problem. Improvement of the accuracy of the constructed solution is possible by running the algorithm several times on a transformed set of points, for example, we can rotate all the points at a certain angle (which is equivalent to dividing the set of points with a line that is not parallel to the X axis). Determining the optimal number of iterations requires further investigation.

# ***Reference***

[1] M. F. Worboys, M. Duckham, “Monitoring qualitative spatiotemporal change for geosensor networks”, International Journal of Geographic Information Science 20(10), 2006, pp. 1087-1108.

[2] A. Galton, “ Dynamic collectives and their collective dynamics”, COSIT 2005, LNCS, vol. 3693, pp. 300-315, 2005.

[3] H.J.Miller, J.Han, J.Eds, Geographic Data Mining and Knowledge Discovery, CRC Press, 2001.

[4] A.Galton, M.Duckham, “What is the region occupied by a set of points?”, GI Science 2006, LNCS, vol. 4197, pp. 81-98, 2006.

[5] S.P.Fekete, On Simple Polygonalizations with Optimal Area, Discrete and Computational Geometry. Springer, 2000.

[6] Generating random simple polygons, http://jeffe.cs.illinois.edu/open/randompoly.html. Last accessed 09 Apr 1999.

[7] T. Auer, M. Held, “Heuristics for the Generation of Random Polygons”, in: 8th

Canad. Conf. Comput. Geometry, pp. 38-44. Carleton University Press, 1996.

[8] M. T. Taranilla, E.O. Gagliardi, G. H. Penalver, “Approaching Minimum Area Polygonization”, in: XIV Workshop de Investigadores en Ciencias de la Computacion, pp. 271-281, 2012.

[9] V. Tereshchenko, V. Muravitskiy, Constructing a simple polygonalizations. Journal of World Academy of Science, Engineering and Technology (77), pp. 668-671, 2011.