Optimizing the Algorithm Localization Mobile Robot Using Triangulation Map

Dao Duy Nam[1,2] and Nguyen Quoc Huy[1,3](✉)

[1] SaigonTech, SaigonTech Tower, Lot 14, Quang Trung Software City, District 12, Ho Chi Minh City, Vietnam
{namdd, huy.nq}@saigontech.edu.vn
[2] High School for the Gifted, VNUHCM, 153 Nguyen Chi Thanh, District 5, Ho Chi Minh City, Vietnam
[3] Saigon University, 273 An Duong Vuong, District 5, Ho Chi Minh City, Vietnam

Abstract. The problem of minimum distance localization in environments that may contain self-similarities is addressed. A mobile robot is placed at an unknown location inside a $2D$ self-similar polygonal environment P. The robot has a map of P and can compute visibility data through sensing. However, the self-similarities in the environment mean that the same visibility data may correspond to several different locations. The goal, therefore, is to determine the robot's true initial location while minimizing the distance traveled by the robot. We consider approximation algorithm for the robot localization problem. The algorithm is based on triangulation of a simple polygon representing a map. Based on the basis of the implemented program, we conducted experimental studies of this algorithm. The numerical results and their interpretation are better than others.

Keywords: Computational geometry · Robotics · Robot localization · Overlay polygon · Algorithm complexity · Approximation algorithm · Polygon triangulation

1 Introduction

In order to solve the application of mobile robot localization problem (MRLP), which relates to the field of robotics using computational geometry methods and algorithms. Substantially, MRLP for the case in a plane is formulated as follows: The mobile robot can move in the external environment, which can be represented as a free space plane and limited by wall (barrier). We assume that the external environment can be described by a simple polygon, the interior corresponding to the free space, and the polygon boundary (without self-intersections) corresponding to the barrier. Assume that the robot is provided an environment map as a simple planar polygon P with n vertices without holes. Initially, the mobile robot is placed in an unknown location at some point p within the polygon P. The robot is equipped with a compass and a sensor device, in which it carries out all-round visibility and the distance to obstacles. Robot must determine its true location in the external environment which locates yourself on the map. For this robot, firstly, it can view their surroundings and realize the visible region (so-called visibility polygon) $V = V(p)$ in the map. If the map has only one piece, which coincides with the visibility polygon, then the problem is solved. If the map has several pieces, it is necessary to determine which piece corresponds to the initial robot location. For this purpose, based on the analysis of polygons P and V, robot must generate a set of hypotheses H of its location $p_i \in P$ so that the visibility polygon $V(p_i)$ at the point p_i is congruent to V. Robot moves and surveys surroundings, then it can eliminate all false hypotheses of its location, and determine its true initial location. This requires that the total length of robot movement must be minimal.

2 Related Works

It proves that the optimization problem of mobile robot localization is an NP-hard problem [1]. The approximation (for polynomial computational complexity) algorithms for mobile robot localization are considered [2–4]. Here, as a rule, it focuses on the characteristics in which describe the deviation value of the optimal solution (the total length of the robot movements). Characteristics of the computational complexity of such algorithms are evaluated asymptotically, and the data of the real time operation algorithms typically are not provided due to the high computational complexity algorithms. For example, $O(n^5 log\ n)$ [2] or $\Omega(n^{12})$ [4]. In [5, 6], they proposed approximation algorithms with MRLP solutions based on the pre-triangulation of a simple polygon representing the map. Now we will discuss the improvements of these algorithms to gain more efficiency in computation time.

3 Solution

The well-known mobile robot localization algorithms [2–4] comprise two phases: hypothesis generation [7] and hypothesis elimination. The hypotheses generation phase computes the set of hypothetical locations $p_1, p_2, \ldots, p_k \in P$ that match the observations sensed by the robot at its initial location. The hypothesis elimination phase rules out incorrect hypotheses thereby determining the true initial location of the robot.

The hypothesis generation phase generates a set $H = \{h_1, h_2, \ldots, h_k\}$, $(\forall i \in 1..k|\, h_i : p{=}p_i)$ of hypothetical locations in P at which the robot might be located initially. Without loss of generality, we select an arbitrary hypothetical location p_i from H to serve as a reference point or origin. Next, for each hypothetical location p_j, $1 \leq j \leq k$, a translation vector $t_j = p_i - p_j$ is defined that translates location p_j to p_i $(p_i = p_j + t_j)$. As a result, we compute a set of copies P_1, P_2,.., P_k of the environment polygon P, corresponding to the set of hypothetical H, such that P_j is congruent to P translated by vector t_j. Copy P_i is translated by the zero vector.

Computing the intersection of polygons is required in algorithms [3, 4], as well as in two further considered algorithms [5, 6]. Note that (1) when computing the shortest path in a simple polygon from a point to another, algorithm [8] uses the provisional triangulation of a polygon followed using “funnel”, this is more effective than other approaches. Constructing the graph visibility [9] is an example; (2) through the using triangulation polygon can be more efficient to compute polygons visibility and its skeletons.

We give a description of the proposed improvement in the mobile robot localization algorithm using triangulation map presented before [6]. Our description has a different specification and a more detailed representation in some steps of the algorithm as well as evaluations of their complexity. In the future, their program will be better if using more systematic triangulation map.

As already mentioned, it is expedient to triangulate the polygon map localization algorithms using an auxiliary effect in case of computing the shortest path from a point in a simple polygon to another. In using the original polygon triangulation preprocessing, firstly, it is possible to effectively implement other action, such as the visibility polygon, and secondly, it is possible to partition the map into many triangles to select the robot movements on the hypotheses elimination phase. For example, a survey of the robot path can occur in the centers of the triangles or middle points of the triangulation edges. It can be considered as an alternative to triangulation on a polygon decomposition visibility cells used in [2, 7]. Number of cell triangulation (triangles) is $O(n)$, which gives a hope for the acceleration of the localization algorithm in using.

Robot localization algorithm using triangulation map [6] can be described by considering the previously given definitions. Let an input polygon map be P and the robot be placed in an unknown initial location in P. The algorithm consists of the following steps:

1. Compute the relative coordinates in the visibility polygon $V(p)$ and its skeleton $V^*(p)$ of the robot according to current sensors in the initial robot location p that the conformance has been not known on the map. Let m be the number of vertices of $V^*(p)$.
2. Make a triangulation of polygon map.
3. Generate set H of k hypotheses on the map P, corresponding to the obtained visibility polygon V. For this polygon skeleton visibility V^* specified in relative coordinates, it is mapped to polygon sequence of vertices based on marking edges skeleton. In places matches based on the use of map which is calculated triangulation skeleton of a polygon of visibility concerning, the alleged location of the robot and the two skeletons are compared. If both skeletons and visibility polygons are the same, then a new hypothesis is fixed.
4. [From P.4 to P.9 in further operations are performed for all active (not yet eliminated) h_j hypotheses ($j = 1, 2,\ldots, k'$), initially $k' = k$.]. Choose an arbitrary hypothesis h_i from a variety of active hypotheses H and the corresponding point of the hypothetical location as a starting point for the construction of intersections. Transferring the vertices gotten from triangulated "displaced" polygon corresponding to other hypotheses (actually in triangulation, the numbers of given vertices do not change, and the "shift" only is occurred in the coordinates of the vertices).
5. For the active hypotheses, the connected component $F = InterS(P_1, P_2, \ldots, P_{k'})$ is calculated at a starting point. Calculate triangulation F.
6. Find the point r in a set of points at the midpoints of edges of the triangulation and at the center of triangles within the polygon F, the point r is the nearest to the

current location of the points of the robot which may eliminate some hypotheses. For this search, the problem is implemented in width by triangulation graph. The visibility polygon skeleton *V**(*q*) is determined in each surveyed of the corresponding point q. This skeleton is associated with the skeletons which appear in the corresponding point q_j in copies map of active hypotheses, and is determined by the possibility elimination of hypotheses when the planned moves of robot are implemented to this point. For all of these points, the shortest path is calculated from the current location of the robot to the selected point *r*. Here we need to calculate the shortest path possible at the stage of the search to get the width of a set of triangles from the current location of the robot to the point *r*, and then immediately to build a funnel to calculate the shortest path. With this modification, it is important that the width of the search tree is not completely bypassed, as cut off subtrees, whose roots are found in terms of possible hypotheses elimination. This search limits in width search, maybe the total movement of the robot is longer, but it reduces the time of the algorithm.

7. Carry out the movement robot to a point *r*.
8. Eliminate hypothesis by comparing the current visibility polygon data of the robot at the point *r*, the data of visibility calculated equivalent in all points corresponding to the active hypotheses. Thus, if a hypothesis is confirmed, it eliminates one or more others, i.e. eventually it eliminates at least one of the hypotheses.
9. Let *E* be a set of hypotheses that have been corrected in the previous step. Replace k' to k' -$|E|$. Repeat steps 4–9 until the set of active hypotheses *H* does not remain only a hypothesis ($k' = 1$), which will correspond to the true initial location of the robot.

The computational complexity of this algorithm in stages and summary are shown in Table 1. Note that when the pre-triangulation calculation of the shortest path from

Table 1. The computational complexity of algorithm in stages and summary

Step	Actions	Complexity of integrated action
1–2	Triangulation polygon map	$O(n \log^* n)$
3,4	Generating hypotheses	$O(mn) + O(kn)$
5	Construction of intersection with respect to the selected hypothesis and triangulation of F (k'- the number of active hypotheses)	$(k' - 1)O(n) + O(n \log^* n)$
6,7	Examine 4 k'(n-2) points on the edges of the triangles and the centers to eliminate hypotheses. Calculation of the shortest paths to the points that eliminate hypotheses to determine the nearest of them	$4k'(n-2)O(n) = k'O(n^2)$
8	Comparison of data on visibility polygons for active hypotheses and current robot location	$k'O(n)$
The full complexity	$O(mn) + O(kn) + O(n \log^* n) + \sum_{k'=1}^{k-1} [k'O(n^2) + k'O(n)] = O(n^4)$	

the current robot location is analyzed in terms of the worst case scenario for the time $O(n)$ [8], but this operation is required to apply repeatedly in step 6 each iteration of the algorithm that determines the total localization algorithm complexity.

Triangulation simple polygon has a theoretical complexity $O(n)$ [10], and can be implemented. For example, a known efficient and practical algorithm [11] for time $O(n\ log^{*}\ n)$ (Let $\log^{(i)} n$ denote the ith iterated logarithm, i.e. $\log^{(0)}n = n$ and for $i > 0$ we have $\log^{(i)}n = \log(\log^{(i-1)}n)$. For $n > 0$ let $\log^{*}n$ denote the largest integer l so that $\log^{(l)}n \geq 1$) practically gives $O(n)$ in MRLP. The output of this algorithm is a set of triangles, the numbers of their given vertices. For further using the algorithm localization, the set of triangles is converted during the $O(n)$ in a special data structure [12]. This view, in fact, is one of the options adapted to the triangulation costal list used to represent a planar subdivision plane [9]. Each triangle in the structure is represented by its three vertices and three-pointers on the adjacent triangles to its adjacent through edges.

Structure triangulation is obtained in two stages. First, a set of given triangles forming a triangulation, and each vertex of the triangulation create a list of triangles in which include the vertex. Then, a proper structure of triangulation with the information about the adjacent triangles is obtained from each pair of the vertices of the triangle by analyzing lists obtained at the first stage. The structure is complemented by inputs array of vertices and edges of the triangulation.

The triangulation structure is used to effectively implement some basic operations of the localization algorithm. For example, it can be systematically used by the operation of construction of the polygon, and the visibility of its skeleton is implemented on the basis of the breadth-first search on the graph by using a dual triangulation. Thus, it is possible, as a rule, to avoid viewing maps all vertices of a polygon, to analyze the triangles adjacent to the current.

4 Implementation

To analyze the effectiveness of the proposed algorithms was carried out an experimental study based on their program implementation (Visual C++ 2010), as well as implementations of algorithms [2–4] and compare their properties as the value of the total travel path of the robot and the running time localization. It has previously been established [13] that the algorithm [3] using the randomization in hypothesis elimination is more efficient than the algorithm [2] using the decomposition map to cell visibilities and having an asymptotic complexity $O(n^5 \log n)$, and also more effective than algorithm [4] based on the solution semigroup Steiner problem and having a computational complexity $\Omega(n^{12})$. Therefore, the comparison is further provided with an algorithm [3]. On algorithms participating in the experiment will be cited, indexed them for brevity Roman numerals: I - mobile robot localization algorithm (MRLA) using triangulation map; II - MRLA randomization using hypothesis elimination [3] (the number of points placed randomly in the test area, $X = 100$); III - the same MRLA [3], but the number of points placed randomly in the study area, $X = 500$; IV - MRLA using windows in the polygon map [5].

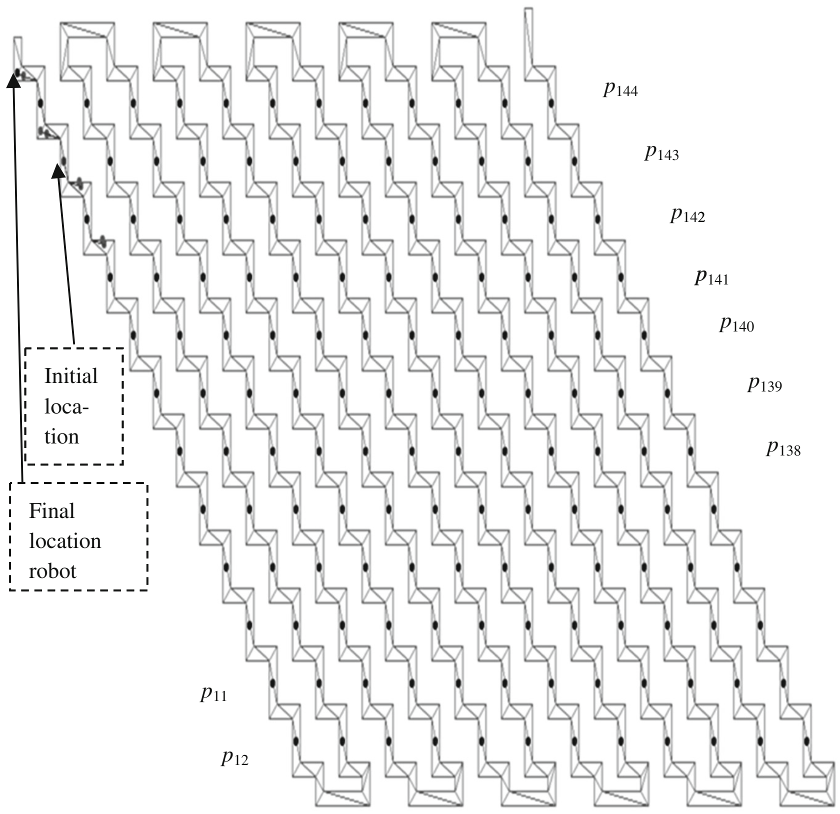

Fig. 1. Generation map $n = 672$ and $k = 144$

Generate maps of various model types used during the experiment. Generation was carried out in several patterns. The combination of the parameters defines the size of the template of the overall size map n, and a generation of its numerical value can be adjusted as a rule approximately. Figure 1 shows an example of generation size map $n = 672$ and $k = 144$ number of hypotheses location p_i ($i = 1..144$). Note that such map structure provides a relatively large ratio of k/n, here $k/n = 0.21$.

Preliminary analysis showed that the resulting characteristics of the algorithms (path length d, traversed by the robot to the final location, and time work t localization algorithm) essentially depend on the initial location of the robot (by hypothesis number). For this reason, it is reasonable to calculate the average of hypotheses, not the characteristics obtained for different algorithms and their relationship. The average will be characterized by the relative efficiency of the algorithms. Choose characteristics of triangulation algorithm (algorithm number I) as the "standard" for comparison. Those algorithms (with a number) will be calculated (here, i - number of hypotheses).

Table 2. The values of the ratios for the configuration shown in Fig. 1.

	Number algorithm			
Feature (x)	I	II	III	IV
The length of the path	1	1.57	1.01	1.63
Time localization	1	17.1	26.7	26.9

$$s_i^{(a)} = \frac{x_i^{(a)}}{x_i^{(I)}}, \text{ where x} = d \text{ or } x = t. \qquad (1)$$

Averaging over k hypothesis was $\bar{s}^{(a)} = \frac{1}{k}\sum_{i=1}^{k} s_i^{(a)}$. Where $\bar{s}_i^{(I)} = 1$ and $\bar{s}^{(I)} = 1$ (Table 2).

The algorithm I based on triangulation shows that the time is much better than that of the others, but it gives a somewhat greater path length than randomization algorithm in variant III. Similar relations are obtained and other map configurations. Figures 2 and 3 show a plot of the mean values of these relations on the size map polygon n. Many types of line graphs in Figs. 2 and 3 are algorithms with their corresponding numbers.

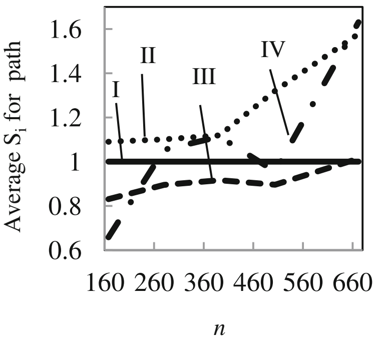

Fig. 2. Average S_i for path

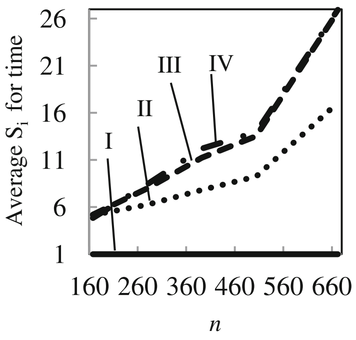

Fig. 3. Average S_i for time

5 Conclusions

The data in these figures show that the range of values the size maps n in the algorithm I above indicates that the operating time is the least, and the second one is the algorithm II.

The experiments with other model configurations maps showed the same results, and our improvement of approximate MRLA algorithms provides comparable accuracy, but the operating time of two modified MRLA algorithms using triangulation map and using windows in the polygon map is smaller than that of others. Here the best algorithm by time results is algorithm I, systematically using triangulation map. This result differs from the previously obtained for the original (unmodified) version of the algorithm [5, 6], using triangulation, where the best results showed the algorithm II at that time.

References

1. Dudek, G., Jenkin, M.: Computational Principles of Mobile Robotics, 2nd edn., 406 p. Cambridge University Press (2010)
2. Dudek, G., Romanik, K., Whitesides, S.: Localizing a robot with minimum travel. SIAM J. Comput. **27**, 583–604 (1998)
3. Rao, M., Dudek, G., Whitesides, S.: Randomized algorithms for minimum distance localization. Internat J. Robotics Research **26**, 917–934 (2007)
4. Koenig, S., Mitchell, J.S.B., Mudgal, A., Tovey, C.: A near-tight approximation algorithm for the robot localization problem. SIAM J. Comput. **39**, 461–490 (2009)
5. Дао Зуй Нам, Ивановский С.А. Приближенный алгоритм локализации мобильного робота с использованием окон в многоугольнике карты. Известия СПБЭТУ «ЛЭТИ» . № 3, С. pp. 38–43 (2014)
6. Дао Зуй Нам, Ивановский С.А. Приближённые алгоритмы локализации мобильного робота. Научный вестик НГТУ. № 2, С, pp. 109–121 (2014)
7. Guibas, L.J., Motwani, R., Raghavan, P.: The robot localization problem. SIAM J. Comput. **26**, 1120–1138 (1997)
8. Hershberger, J., Snoeyink, J.: Computing minimum length paths of a given homotopy class. Comput. Geom. Theory Appl. **4**, 63–98 (1994)
9. De Berg, M., Cheong, O., Van Kreveld, M., Overmars, M.: Computational Geometry: Algorithms and Applications, 3rd edn., 386 p. Springer, Heidelberg (2008)
10. Chazelle, B.: Triangulating a simple polygon in linear time. Discrete Comput. Geom. **6**(1), 485–524 (1991). Springer-Verlag
11. Seidel, R.: A simple and fast incremental randomized algorithm for computing trapezoidal decompositions and for triangulating polygons. Comput. Geom. Theory Appl. **1**(1), 51–54 (1991)
12. Скворцов А.В. Триангуляция Делоне и её применение. – Томск: Изд-во Том. ун-та, 2002. – 128 с
13. Дао Зуй Нам, Ивановский С.А. Экспериментальный анализ алгоритмов локализации мобильного робота. Известия СПБЭТУ «ЛЭТИ» . № 1. С, pp. 19–24 (2014)