British Journal of Mathematics & Computer Science
17(1): 1-14, 2016, Article no.BJMCS.26350
ISSN: 2231-0851

SCIENCEDOMAIN *international*
www.sciencedomain.org

Construction of Picture Maze along Set of Image Dot Vertices

Tomio Kurokawa[1*]

[1]*Department of Information Science, Aichi Institute of Technology, 1247 Yachigusa, Yagusa-cho, Toyota 470-0392, Japan.*

Author's contribution

The sole author designed, analyzed and interpreted and prepared the manuscript.

Article Information

DOI: 10.9734/BJMCS/2016/26350
Editor(s):
(1) Victor Carvalho, Polytechnic Institute of Cávado and Ave, Portuguese Catholic University and Lusiada University, Portugal.
Reviewers:
(1) Yogesh Meghrajani, Dharmsinh Desai University, India.
(2) K. Manikandan, Sri Krishna College of Engineering & Technology, Coimbatore, India.
Complete Peer review History: http://sciencedomain.org/review-history/14844

Original Research Article

Received: 12th April 2016
Accepted: 24th May 2016
Published: 31st May 2016

Abstract

There are three interests in this study. First is to make an analogy between graph theory and image processing. Given a number of image items (dots) on a picture, this paper shows a way to make a path (closed path) connecting and traversing all of them but each only once, imitating to construct a path (closed path) on a plane by considering a dot as a vertex and a connecting line as an edge. This looks like constructing a Hamiltonian path on a plane and drawing a plane graph connecting all vertices with no crossing among the edges. Second is to construct the maze based on the obtained path with some detouring within each of image items (dots). This path could be very long and complex. Owing to this path, the maze is constructed in an organized manner. Third is to apply this path making procedure to different pictures with different image shapes like symbols or English letters --- possibly generating a complex picture maze. Pictures or some meaningful message appear when the maze is solved. Algorithm and its proof are given with a number of successful experimental results.

Keywords: Maze; picture maze; graph theory; path; Hamiltonian path; plane graph; image processing.

1 Introduction

A maze in this study is a square maze organized in a computer array [1]. It consists of paths and walls. Path consists of a set of vertices and a set of edges like a graph. But more restraints are imposed on the maze.

**Corresponding author: E-mail: kurokawa@aitech.ac.jp;*

Vertex of the maze is placed one in every two cells vertically as well as horizontally in the array. If two adjacent vertices are connected, then there is an edge between the two and all the vertices in the array are connected but no cycle is allowed. So the maze is a spanning tree on the array. If two vertices on the tree are designated as the start and the goal, then the path between the two is the solution.

A maze constructed in this study is based on the maze structure above. But it is generated based on an original binary picture. Employing techniques of image processing, graph theoretical concepts, and following the constraints of the maze structure, a kind of complex maze (picture maze) is generated. Showing a part of the experimental results would rather work better to make the introduction of this study and should be more persuasive.

Fig. 1 is the original picture from which a maze would be constructed. Through some processes, a maze such as in Fig. 2 can be obtained. However, it is the constructed maze with the solution path (red) painted on it. If the dots have some shapes, then some pictures could appear accordingly.

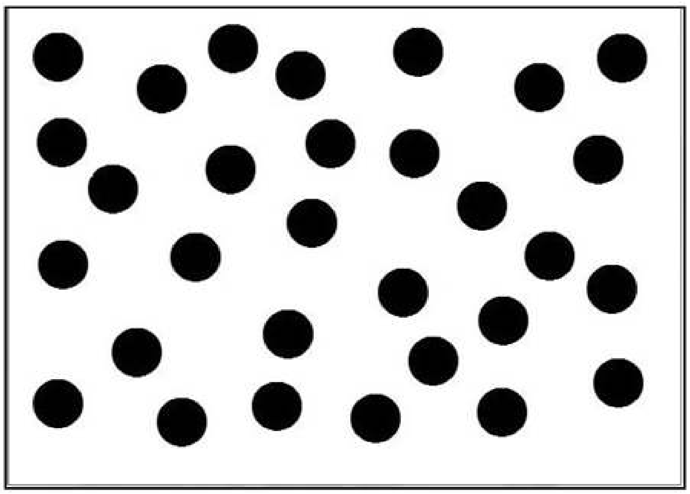

Fig. 1. Original picture of image dots; number of dots=30

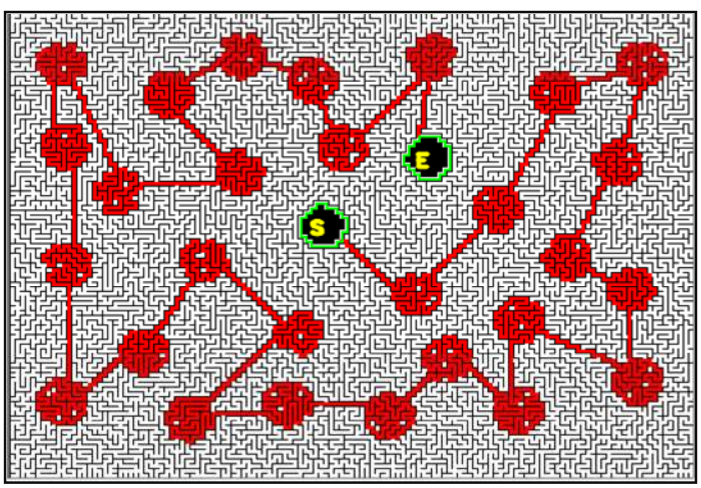

Fig. 2. Constructed maze of image dots with solution

2 Literature

Maze generation by computing on square grids has been around for some years. The article [2] showed various algorithms to generate a square grid based maze, usually used, relating with graph theory. Hosokawa [1] presented a number of maze generation algorithms with the relation between path, wall, and coordinates. Xu and Kaplan [3] illustrated complex mazes emphasizing aesthetics for the maze. They also introduced a picture maze [4] called 'Maze-a-pix' by illustrating a sample picture maze, which is provided by Conceptis, Ltd. [5]. However, the method of the maze generation was not given. The internet site of the company shows various picture mazes and states a history of computer generated picture maze. Okamoto and Uehara [6] demonstrated an automatic generation algorithm for a picture maze by constructing a Hamiltonian path which fills up the picture region. However, the path is generated on the 2-by-2 extended array. The maze is so constructed that the player obtains the solution path by filling up all the path cells on the maze and by going through the path. Hamada [7] improved the method [6] by enabling to place the start and goal at different positions. Kurokawa developed a more efficient algorithm of construction of Hamiltonian path with a rigorous proof to generate a picture maze and introduced a generalized idea for maze path construction [8]. Ikeda and Hashimoto [9] illustrated a picture maze generation method by using simulated annealing. Kurokawa, Mori, and Mizuno [10-12] took a different approach and demonstrated the method to generate a picture maze by making use of the picture contours. In the method, all the picture contours are extracted and are incorporated into the maze solution path. It is a very flexible method and can be applicable to a wide range of binary pictures. F. J. Wong and S. Takahashi [13] constructed a hybrid picture maze from two photograph images. This maze is not based on the square grid type mazes but on the special grids computed from the image. Two photo pictures are overlapped in the maze. The maze player depicts an outline of one picture on the other picture background.

Although not about the picture maze construction, the study [14] is the one that makes a typical sample for relating the graph theory and the image processing. It extracts a Euler Circuit from one stroke drawing of a closed path by making use of the contours of the drawing and generates the maze solution path by applying graph theory including by-partitioning to the set of the contours. The present study itself also is a try of making an analogy between graph theory and image processing---realizing a Hamiltonian-like path by connecting image items on an image plane.

3 Preceding Studies

This study is about one of picture maze generation methods. Graph concepts were incorporated in image processing and maze construction. There are a number of preceding studies which have made some influences to this study. First is the one that made a picture maze by merging image contours [10-12]. In the study, image contour cycle extraction and contour merging played important roles. The second is to generate Hamiltonian-like path within an image item so that some image or picture appears when the maze is solved [8], features of which mainly brought a hint for doing the research on the path construction in this study. The third is Eulerian circuit, organized from a closed path of one stroke drawing [14].

4 Concepts of this Study

The purpose of this study is three-fold. First is to examine if it is possible to construct a closed path which traverses all the image dots, usually of large number (Fig. 1), each once, which are randomly scattered on a plane. This is to make an analogy between image processing on image plane and graph theory and generate a closed path-like structure on an image array. An image dot (Fig. 1) is considered as a vertex of a graph and a connecting line between dots as an edge. This problem is also considered as drawing a closed path (a plane graph) traversing all the vertices on the plane with no edge intersection.

As stated, second is to construct the complex maze from the image dots such as in Fig. 1. Fig. 2 is the completed maze with the solution path drawn as red. Starting from ‘S’ circled, connecting all the image dots, and ending at 'E'. This path looks like a Hamiltonian path, which traverses all the image dots (vertices of a graph), each dot once. In addition, a Hamiltonian-like path is also constructed within every image dot. Hence the entire path could be fairly complex.

Third is to get a picture maze by applying the method to different pictures with some shaped items replacing some image dots. It enables the shapes to appear when the maze is solved.

4.1 Image dots as contour cycles and path making

Fig. 1 is the typical original picture from which a complex maze is constructed. The program with VC++ for this study first recognizes each of the image dots of such pictures by extracting the contours of the dots, which form cycles and are numbered by OpenCV library from 1 to n (=30) as shown in Fig. 3. The cycles are organized with the maze path structure which is explained in section 1 and the previous works [8,10-12]. The connections between image dots are all based on these cycles. There are *n* dots in the picture. Then there should be *n* cycles with each having n_j length (number of vertices and edges for the cycle). A cycle, $cycle_j$, is a sequence of coordinates and expressed as:

$$cycle_j = \{ \left(x(j,i), y(j,i)\right) \mid 1 \le i \le n_j\} \tag{1}$$

The set of all cycles, CYCLES, extracted is expressed as:

$$CYCLES = \{ cycle_j \mid 1 \le j \le n\} \tag{2}$$

Each cycle is considered as a vertex of graph:

$$cycle_j = v_j \tag{3}$$

There may be a number of ways to make a closed path --- connecting all the dot cycles. One approach may be to connect one to a near one repeatedly until all are connected. This paper proposes a method to make a closed path connecting all the cycles once for each with no intersection between connecting lines. This means that the path drawn is a plane graph. Details are described in the following subsections.

4.2 Algorithm

Input: A set CYCLES of *n* image dot cycles by (2), Fig. 3; each cycle is considered as a vertex (3).

Output: A closed path P of length *m* (initially 0 and *n* at the end) with no intersection among the connecting lines (edges). P is comprised of a set of ordered cycles (4). But it represents an ordered set of edges. The last edge is by $cycle_n cycle_1$.

$$\mathrm{P} = cycle_1 cycle_2 \cdots cycle_n \tag{4}$$

Each cycle is in turn considered as a vertex. So P is an ordered set of vertices (5).

$$\mathrm{P} = v_1 v_{2 \dots} v_n \tag{5}$$

Definitions:

The distance from $cycle_j$ to $cycle_k$ or v_j to v_k is defined as:

$$d(j,k) = min_{1 \le p \le n_j \,\&\, 1 \le q \le n_k} \sqrt{\left(x(j,p) - x(k,q)\right)^2 + \left(y(j,p) - y(k,q)\right)^2} \tag{6}$$

P(j) is the *jth* item in P, which is circular.

Method:

1. Select an arbitrary $cycle_i$ from CYCLES; and delete it from CYCLES.
2. Find the nearest cycle from *i* and name it *j* and delete it from CYCLES.
3. Find $cycle_k$ from CYCLES so that $d(j,k) + d(k,i)$ becomes minimal with no cross among connecting lines and cycles.
4. Organize P with constituent cycles *i, k, and j* which are $v_i, v_k, and\ v_j$ and order them as 1, 2 and 3. Let the length of P be *m* (=3);
5. If CYCLES becomes empty, algorithm terminates with *m=n* and draw P.
6. For all *j* from 1 to *m* and all *q* in CYCLES do {
    If(*j==m*) {*a=j* and *b=1*} else {*a=j; b=j+1*;}
    Find *a, b* and *q* so that $d(P(a),q) + d(q,P(b))$ becomes the minimal
      with no cross among connecting lines and cycles.
    }
  }
7. Insert *q* between P(a) and P(b), increment *m* and delete *q* from CYCLES. Go to 5.

Condition:

It is assumed that the sizes and the shape of contour cycles do not interfere the connections among them. This is usually justified.

Explanation:

Since $d(P(a),q) + d\big(q,P(b)\big)$ is the minimal, no cycle exists inside the triangle $(P(a),q,P(b))$, hence inside the region surrounded by P. This is explained later. Even if $d(P(a),q) + d(q,P(b))$ is the minimal, there is a possibility that the connecting lines intersects. So selection of *q* is always made under the condition that no intersection occurs. It is naturally justified that at least one contour cycle always exists to be inserted to P if CYCLES is not empty because P is the closed path, circular hence faces in all directions.

Fig. 4 shows that an arbitrary $cycle_{16}$ is selected and $cycle_{22}$ is found as the nearest and they are connected. Fig. 5 shows that $cycle_{19}$ is selected because $d(16,19) + d(19,22)$ is the minimal and form P($cycle_{16}$, $cycle_{19}$, $cycle_{22}$) as clockwise. Fig. 6 illustrates that $cycle_{26}$ is selected and inserted in P and the path becomes P($cycle_{16}$, $cycle_{19}$, $cycle_{26}$, $cycle_{22}$). Fig. 7 is the final closed path which visits all cycles, each only once without intersection.

No attempts were made to seek the efficiency of the algorithm in this study.

Algorithm proof:

The algorithm proof is given by induction:

When there are three (*i*==3) cycles in P, P is C_3, a triangle with no cycle inside. Suppose that P is C_r and no cycle inside the closed path when $i = r$. As long as CYCLES is not empty, there is a cycle in CYCLES, named as *q* such that $d(P(a),q) + d(q,P(b))$ becomes the minimal with no intersection. This is guaranteed because P is circular so it faces in all directions. The cycle *q* is the (*i=r*+1)th cycle deleted from CYCLES and it is the (*r*+1)th cycle inserted in P. Therefore, P becomes C_{r+1}. This completes the proof:

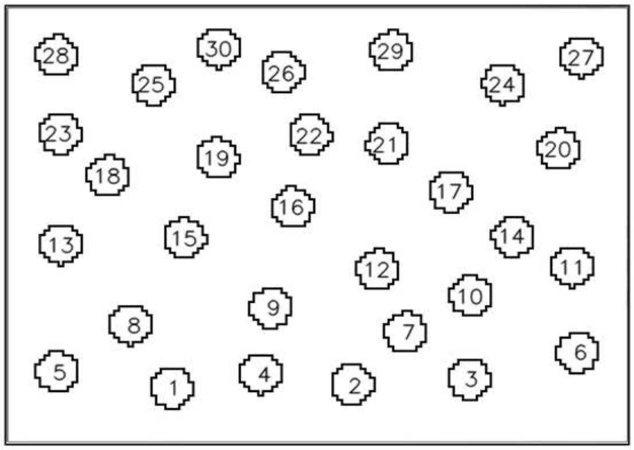

Fig. 3. Dot contour cycles

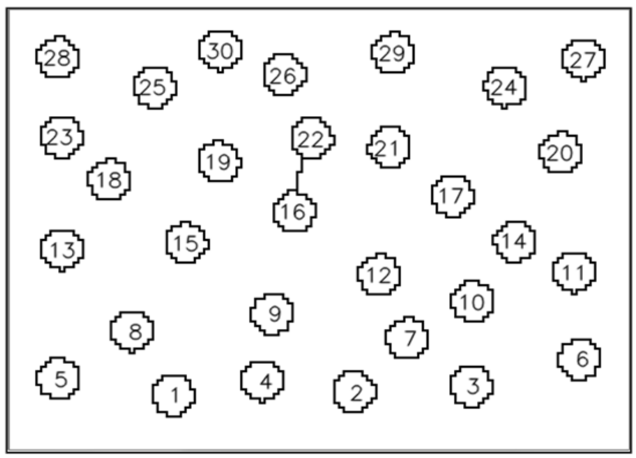

Fig. 4. First connection

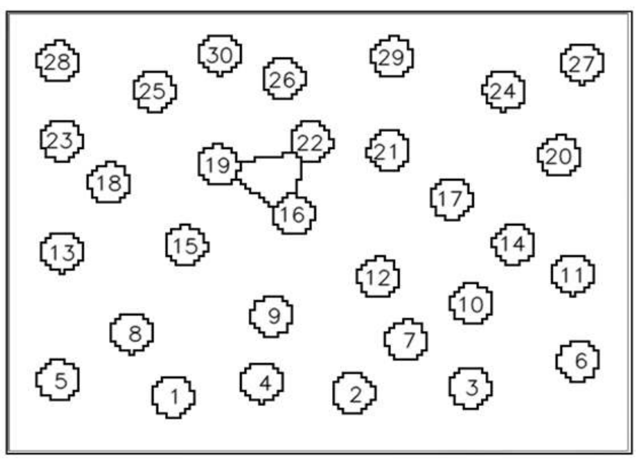

Fig. 5. Three dot cycle closed path

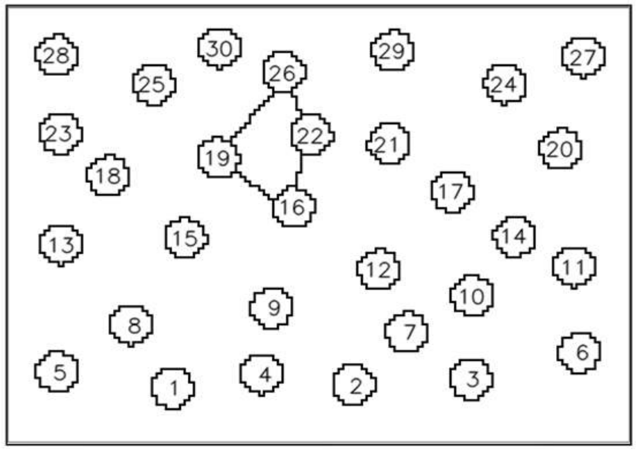

Fig. 6. No. 26 is inserted in the closed path

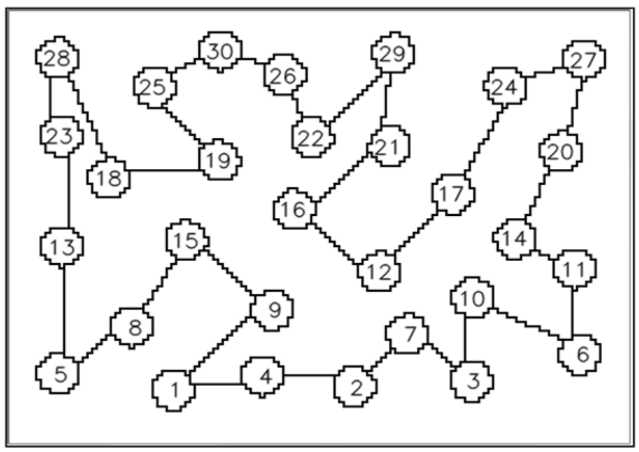

Fig. 7. Closed path completed

5 Obtaining Complete Maze and Discussion

5.1 Maze construction on a number of image dots

Fig. 7 shows the completed construction of the closed path. As seen, the line between dots 18 and 28 became very close to the dot 23. The program is designed just to avoid the collision between the connecting lines and the dot cycles. This kind of narrow collision possibly occurs. However, such collisions between lines are mostly prevented by the criterion introduced in the algorithm, the minimal sum of two edge line lengths of the triangle, explained in the previous sub-section. In addition, this criterion makes each of the connecting curves become shorter. The shorter those lines, the smaller the possibility of the intersection between those and the connecting lines in later cycles becomes. Another effect of the criterion is that the overall path length become rather shorter, if not the shortest. It probably makes the appearance better.

After constructing the closed path (Fig. 7), the following steps were applied. The line between two cycles (16, 21) was removed to designate the start and the goal. The path from the start cycle (16) to the end (21) became one stroke line. Whenever a cycle was encountered going along the path, an arc-like curve was randomly chosen. The result is shown as in Fig. 8. Within an image dot region, Hamiltonian-like path was organized. This was done by successive insertions of path parts using the method [8,9,13]. Almost all points (vertices) within an image dot were inserted into the path, but not necessarily all. This is why white points appear at a number of places in dot vertices (Fig. 9). At this point, the solution path of the maze was competed. Then, random dead end branches were added in order to hide the solution path. Much attention was paid to hide it better. Walls of the maze were thinned to finish the maze (Fig. 10). Fig. 2 is the maze with the solution path drawn red as mentioned before. Fig. 11 is another completed maze with the solution. The connection lines in Fig. 11 were randomly curved a little. This is to hide the solution more effectively.

Fig. 12 is another original picture of image dots. Five more dots were added then the picture frame became almost full. Fig. 13 is the completed maze painted with red solution path. The start and end positions were changed from Fig. 11. The program is so designed that the path construction can start from any of the dot vertices. The overall path structure and the start and end positions may become different depending on the experiments.

5.2 Construction of picture maze

The same procedure was applied to the image items of Fig. 14. The obtained maze is a complex picture maze. Fig. 15 shows the maze with the red solution path. This maze is a quiz to find an English word “COMPUTER”. Accordingly, when the player draws the solution path, letters appear---‘C’, ‘R’, ‘O’, ‘E’, ‘U’, ‘P’, ‘T’, and ’M’. The player is naturally asked to find an English word, “COMPUTER” using the found letters. The connecting lines between image items are with some randomized zigzag so that the lines on the maze cannot be seen easily before it is solved. In order to make the start and end vertices by image dots, a number of dot vertices are placed among English letters. Because of some non-connected points in the image regions, some alphabets do not appear aesthetically. However, we can recognize them.

For some letters, there are holes in the letter such as ‘P’. They may cause a problem. To cope with them, a mask of the original picture was used. And the path extension within letter regions are made with this mask so that the path would not be extended in the hole regions.

Fig. 16 is another original picture with different font letter images. Fig. 17 is the maze quiz with the red solution path. The answer is “SOLVE MAZE.” Since there are not many pairs of image dots, many different path constructions are not possible as long as the start and end are by a pair of the dots.

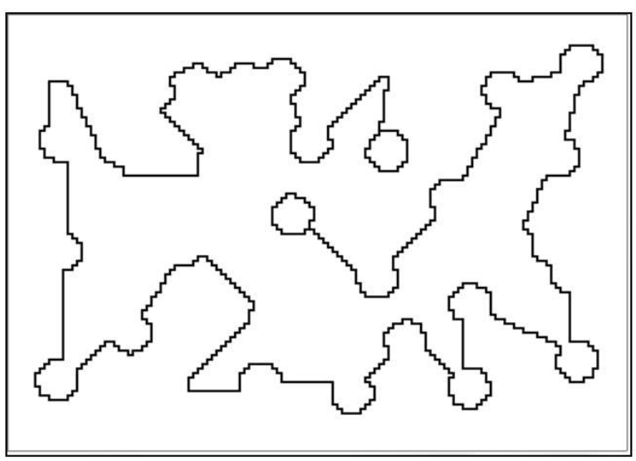

Fig. 8. Starting base line of solution path

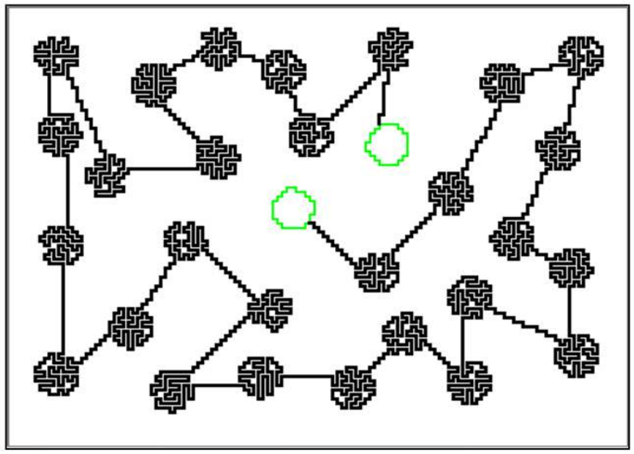

Fig. 9. Maze solution path completed

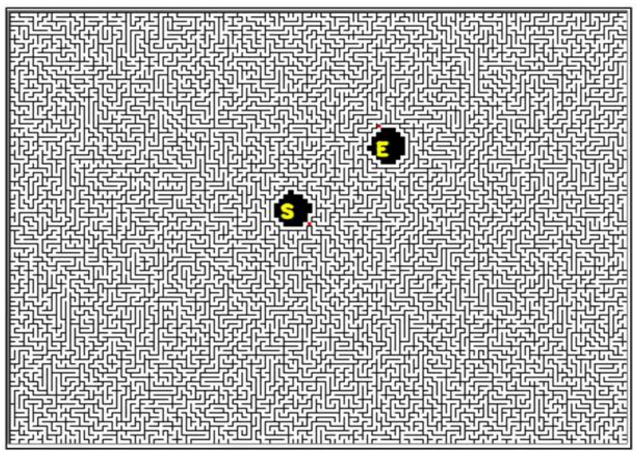

Fig. 10. Final maze completed to be presented to players

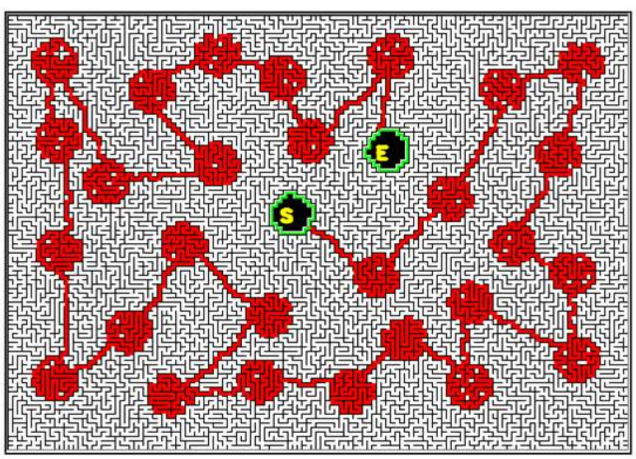

Fig. 11. Final maze but with curved solution path lines

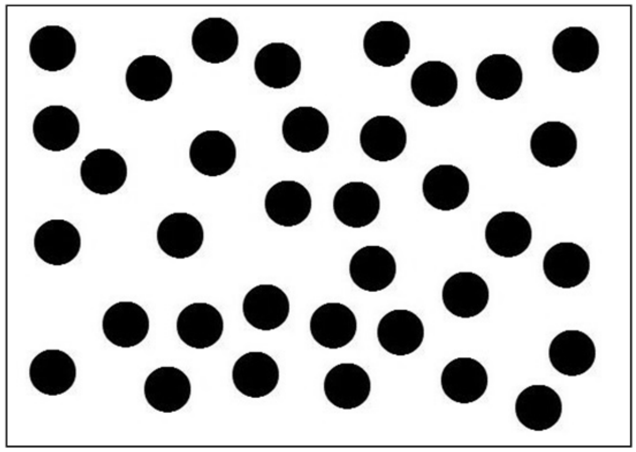

Fig. 12. Original picture of image dots; number of dots=35

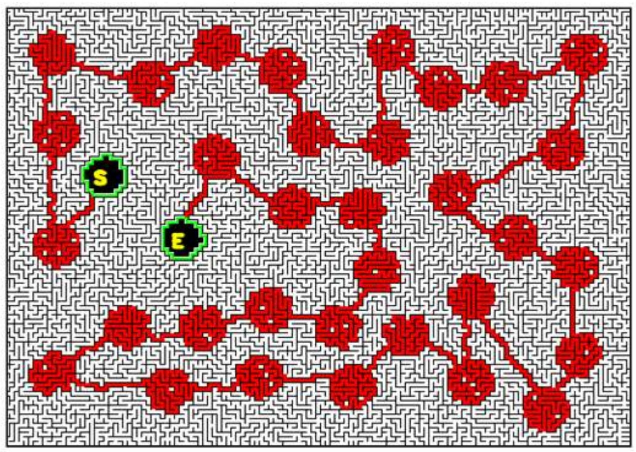

Fig. 13. Final maze with curved solution path lines

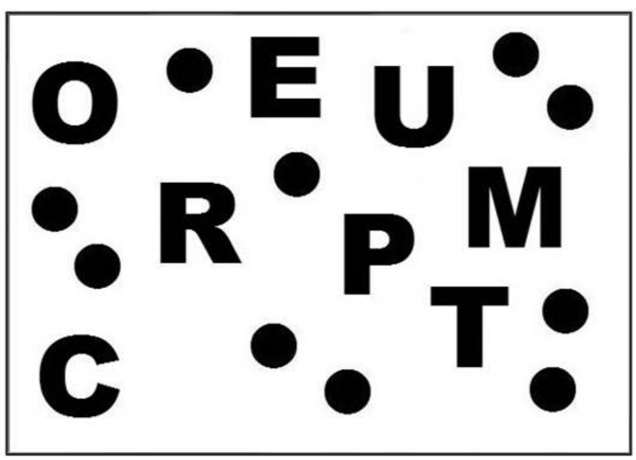

Fig. 14. Original binary picture with the alphabets of "COMPUTER"

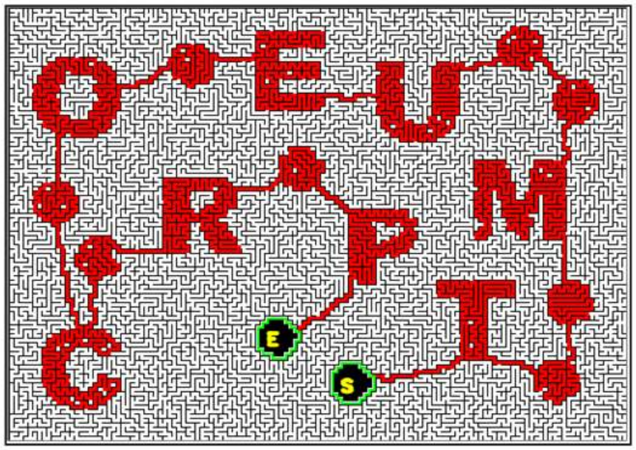

Fig. 15. Complete quiz maze with solution path to find "COMPUTER"

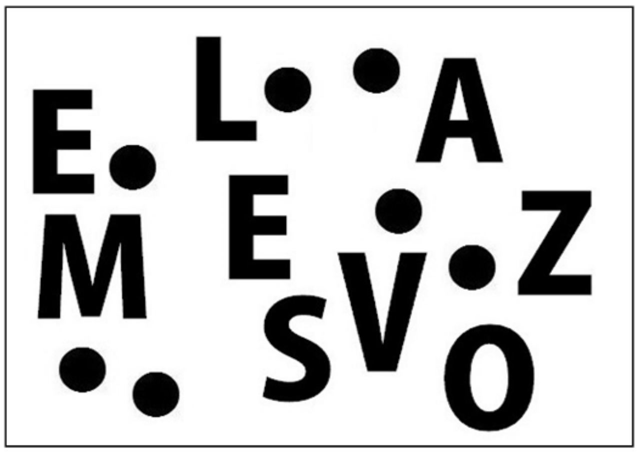

Fig. 16. Original binary picture with the sentence of "SOLVE MAZE"

Fig. 17. Complete quiz maze with solution path to find "SOLVE MAZE"

5.3 About the process of organizing P

P is the closed path of a sequence of the vertices which traverses each of the vertices extracted from a given picture once for every vertex. Whenever one vertex is inserted in P by the algorithm previously stated, there appears a situation illustrated in Fig. 18. Consider the triangle organized by the three vertices $P(a)$, q, and $P(b)$ as in Fig. 18, where q is a vertex in CYCLES and b is the next item number of a in P and the triangle gives the minimal sum of two edge lengths $d(P(a), q)$ and $d(q, P(b))$. If there is a vertex q' inside the triangle from CYCLES, then the following inequality, which is well known, holds.

$$d(P(a), q) + d(q, P(b)) \geq d(P(a), q') + d(q', P(b)) \quad (7)$$

This means that there is no vertex inside the triangle if $d(P(a), q) + d\big(q, P(b)\big)$ is the minimal for a, b in P and q in CYCLES. If the combination of a, b and q always provides the minimal, then there is no vertex inside the region surrounded by P. Therefore, we can continue to look for such vertices outside the region until CYCLES becomes empty.

In case there are more than one vertex satisfying the minimal condition, selecting one of those will not make any problem. The one not selected will be incorporated into the path in later cycles. The criterion of the minimal rather seems to make the connecting lines shorter. If those are shorter, then it reduces the possibility that later connections will make the crossing with those lines.

By the way, we have the following inequality:

$$d(P(a), q) + d\big(q, P(b)\big) = d(P(a), q) + d(q, r) + \ d(r, P(b)) \quad (8)$$

$$\geq d(P(a), r) + d(r, P(b)) \quad (9)$$

$$\geq d(P(a), q') + d(q', r) + d(r, P(b)) \quad (10)$$

$$\geq d(P(a), q') + d(q', P(b)) \quad (11)$$

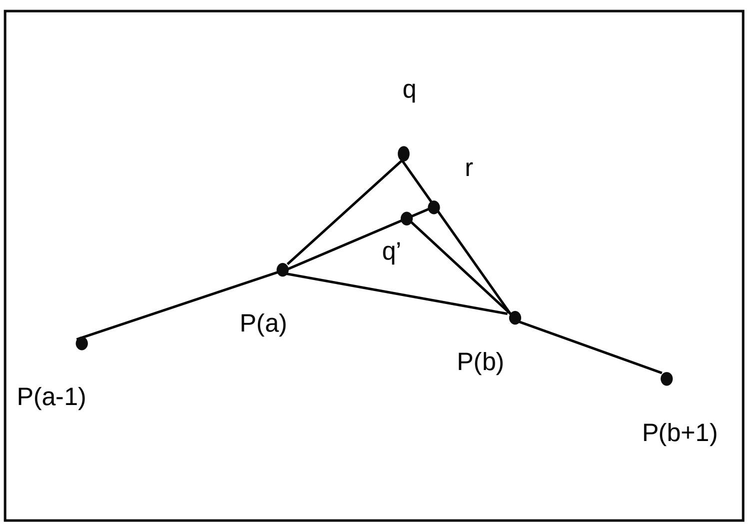

Fig. 18. Triangle ($P(a)$, q, $P(b)$). b is the next item number of a in P which is circular

5.4 Point of this study

Picture maze construction with near Hamiltonian path appears in the studies [8]. Here, 'near' means that the path passes through nearly all the vertices in a set, but possibly not all. The methods in those studies mainly deal with only one or a few picture items using the near Hamiltonian path making method. The present study also employs the same or similar methods in the near Hamiltonian path making within individual picture items. As long as these processes only are concerned, there are no noticeable differences between the present study and [8] in relation with the procedure and the experimental results. However, the present study, in addition to the above, has the ability to handle a large number of picture items in an organized and flexible manner so that it can make a more complex picture maze.

6 Conclusion

The following are the conclusions:

1. Analogy is naturally made between image processing and graph theory or between image space and graph space.
2. The proposed algorithm showed that it constructs a path traversing pre-positioned image dots, possibly large number, once for each. The path looks like a Hamiltonian path. Maze is to be constructed along those image dots as well as the path in an organized manner. The construction algorithm is proved by induction.
3. It is shown that a complex maze can be constructed using the method. If shaped images are used, the maze becomes a picture maze.

Competing Interests

Author has declared that no competing interests exist.

References

[1] Hosokawa T. Maze.
Available:http://aanda.system.to/
(Accessed 2016/05/6)

[2] Maze generation algorithm.
Available:http://en.wikipedia.org/wiki/Maze_generation_algorithm
(Accessed 2016/05/6)

[3] Xu J, Kaplan CS. Image-guided maze construction. ACM Transactions on Graphics. 2007;26(3): Article 29:1-9.

[4] Xu J, Kaplan CS. Vortex maze construction. Journal of Mathematics and the Arts. 2007;1(1):7-20.

[5] Conceptis puzzles. Maze-a-PiX.
Available:http://www.conceptispuzzles.com/
(Accessed 2016/05/6)

[6] Okamoto Y, Uehara R. How to make a picturesque maze. Proc. Canadian Conf. on Computational Geometry. 2009;137-140.

[7] Hamada K. A picturesque maze generation algorithm with any given endpoints. Journal of Information Processing. 2013;21(3):393-397.

[8] Kurokawa T. Picture maze generation by successive insertion of path segment. British Journal of Mathematics and Computer Science. Sciencedomain International.
2014;4(24):3444-3463:12.
ISSN: 2231-0851
Available:http://www.sciencedomain.org/issue.php?iid=699&id=6

[9] Ikeda K, Hashimoto J. Stochastic optimization for picture maze generation. Jouhou Shori Gakkai Ronbunshi. 2012;53(6):1625-2634. (Japanese).
Available:https://dspace.jaist.ac.jp/dspace/bitstream/10119/10912/1/18003.pdf
(Accessed 2016/5/6)

[10] Kurokawa T, Mori K, Mizuno T. Automatic construction of picture maze by repeated contour connection. Thailand-Japan Joint Conference on Computational Geometry and Graphs. 2012;5-6.

[11] Kurokawa T. Picture maze generation and its relation to graphs. 2013 Shikoku-Section Joint Convention Record of the Institute of Electrical and Related Engineers. 2013;231.

[12] Kurokawa T. Picture maze generation by repeated contour connection and graph structure of maze. Computer Science and Engineering. Scientific & Academic Publishing, USA.
2013.12;3(3):76-83.
ISSN: 2163-1484
Available:http://article.sapub.org/10.5923.j.computer.20130303.04.html
(Accessed 2016/05/6)

[13] Wong FJ, Takahashi S. Flow-based automatic generation of hybrid picture mazes. Computer Graphics Forum. Wiley-Blackwell. 2009;28(7):1975-1984.

[14] Kurokawa T. Maze construction by using characteristics of eulerian graphs. British Journal of Mathematics & Computer Science. Sciencedomain International, UK. 2015;6:(9)6:453-472.
ISSN: 2231-0851
Available:http://sciencedomain.org/issue/1147

Peer-review history:
The peer review history for this paper can be accessed here (Please copy paste the total link in your browser address bar)
http://sciencedomain.org/review-history/14844