

A Simple and Unified Method for Automatic Generation of Picture Maze

ABSTRACT

Picture maze is a maze on which some picture appears when it is solved. This paper aims to present a very simple method to construct a picture maze. It is always possible to construct a spanning tree on a connected region of a binary picture. A spanning tree connects all vertices on the region. A 2-by-2 extended picture of a connected picture is the one such that each cell in the picture is transformed to four cells which constitute a square block with the cells. Then by analogy, it is always possible to construct a block spanning tree on the extended picture region, where a block is the square unit of four cells. Construction of the block spanning tree and the generation of the corresponding Hamiltonian path can be done at the same time. The Hamiltonian path is constructed along a data structure of a vector in a simple manner. It traverses every cell on the 2-by-2 extended picture. The similar procedure but with the smaller block with two vertices, instead of four, can be applied to a non-extended picture, an ordinary one, which produces a near Hamiltonian path, which can then be a maze solution path. The near Hamiltonian path traverses nearly all the vertices on the picture, hence depicting a picture on the maze when it is solved. This paper demonstrates the method and its efficiency.

Keywords: Maze, Picture Maze, Spanning Tree, Block Spanning Tree, Path Extension, Hamiltonian Path

1. INTRODUCTION

Maze generation by computing with square grids has been around for some years. The article [1] showed various algorithms to generate a square grid based maze, usually used, relating with graph theory. Hosokawa [2] 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 picture 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 picture array. The maze is so constructed in

the research that the player obtains the solution path by filling up all the path cells on the maze and by painting the picture region. Hamada [7] and [8] improved the method [6] by enabling to place the start and goal at different positions. 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 a method to generate a picture maze by making use of the picture contours. In this method, all the contours are extracted and are incorporated into the picture maze. It is 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 square grids like the 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.

2. MAZE STRUCTURE

There are a number of different representation structures of square grid maze. This paper uses one of them and it is explained in this section. A computer constructed maze is usually represented by a two-dimensional array. Such a maze consists of path and wall. A path cell is either a vertex or an edge (of a graph). Fig. 1 is a sample maze array of size 7x7---where '⊙' is a vertex; 'e' is an edge, and 'W' and 'w' are walls. A vertex is placed one in every two cells horizontally and vertically. An edge represents the connection between the two vertices. Diagonally adjacent cells to a vertex are wall cells; there are four such cells around a vertex; the vertically or horizontally adjacent cells to a vertex are edge or wall cells. If the cell represents the connection between the two vertices, then it is an edge cell; otherwise it is a wall cell.

The connection between the two vertices is 4-way, vertical or horizontal; and no diagonal connection is allowed. The entire path routes organize a spanning tree. A vertex and an edge alternately appear along a path. The maze solution of Fig. 1 is the path from the start (S) to the goal (G), which is one of the paths from the tree root to a vertex. The maze structure is kept maintained even if the maze is shifted in any direction one or more positions on the array. However, in Fig. 1, vertices are placed on (odd, odd) coordinates.

If the walls are thinned [14], the maze becomes as presented in Fig. 2. This maze, to be presented to a player, virtually consists of two kinds of cells, path and wall. However, it represents the corresponding graph with vertices and edges, which cannot be otherwise represented as simply as this structure. So the maze of this structure is said to be an elaborately organized graph. The edges play very important roles in this elaboration. Without the edge cells, additional information is required to show how vertices are connected [15]. In short, a maze is said to be a collection of vertices and edges like a graph. However, more restrictions are imposed on a maze: physical positions of vertices and edges; the number of edges to be connected to a vertex and the like.

	1	2	3	4	5	6	7
1	S	e	⊙	w	⊙	e	⊙
2	w	W	e	W	e	W	w
3	⊙	e	⊙	e	⊙	e	⊙
4	w	W	e	W	w	W	w
5	⊙	e	⊙	e	⊙	e	⊙
6	e	W	e	W	e	W	e
7	⊙	w	⊙	w	⊙	w	G

Fig. 1 Maze structure

	1	3	5	7
1	S			
3				
5				
7				G

Fig. 2 Maze to be presented to a player

3. RELATED PRECEEDING STUDIES

Okamoto and Uehara [6] presented a method to generate a picture maze by constructing a Hamiltonian path on the 2-by-2 extended picture. The 2-by-2 extended picture array is constructed by dividing each cell of the original picture array by two both vertically and horizontally. So each cell of original picture becomes a square unit of four cells in the 2-by-2 extended picture. The Hamiltonian path covers all the vertices on the foreground picture region. So, when the maze player traces the path by filling up all the vertices on the picture region, the hidden picture appears. The reason to use the 2-by-2 extended picture is to guarantee the existence of the Hamiltonian path on the region. The studies of [6] by Okamoto and Uehara and [8] by Hamada explain the algorithm of the maze construction. The algorithm outlines as:

- (1) Prepare a binary picture on square grids;
- (2) Construct a spanning tree on the picture region;
- (3) As traveling around the spanning tree by using the right (left) hand rule, construct the Hamiltonian path on the 2-by-2 extended picture;
- (4) The obtained Hamiltonian path is the solution path of the maze.

4. PROPOSED SIMPLE METHOD

All experiments in this paper were done by using the software [16] and [17].

4.1 Constructing a Spanning Tree

As known, it is always possible to draw a spanning tree on a connected region which is on an array of square grid cells. That is, with repeated random extension of a vertex from one starting vertex, an adjacent vertex can be repeatedly connected to the already connected part until no unconnected vertex exists in the region. A spanning tree completes at the end. Fig. 3 (a) to (d) illustrates how the tree grows and becomes a spanning tree. In Fig. 3, the larger square is the picture region, where the tree can grow. The red cell with a green dot is the vertex and the darker gray cell is an edge. Staring with the red cell with a black dot, the tree grows as the random connection of an adjacent vertex to the tree proceeds. The connection is made with the edge (dark gray cell). Fig. 3 (d) is the final spanning tree.

4.2 Construction of Block Spanning Tree and the Hamiltonian Path

This subsection describes how to construct a Hamiltonian path which fills up all the vertices on the 2-by-2 extended picture simultaneously building the block spanning tree. It is by the analogy of the method of the spanning tree construction in the previous subsection 4.1.

Each cell in the original picture corresponds to the block of four cells in the 2-by-2 extended picture. This provides the principle that if a vertex extension on the original picture to construct a tree is possible, then the corresponding extension of the block of four vertices on the extended picture is possible. If the extension is repeated until all blocks are connected, a spanning tree with blocks is to be completed on the picture region at the end. This means that it is always possible to construct such a spanning tree on the extended picture. The block trees so generated are shown in Fig. 4 (a)-(d), where, they are with the simultaneously being generated path which traverses all the vertices of the block trees without repetition of any vertices or edges. Fig. 4 (d) illustrates the block spanning tree as well as the final Hamiltonian path on the picture region.

Fig. 4 (a) shows the starting block. The extension could be made to three directions: to the right, to the top or to the left. The extension to the bottom is not allowed because it is out of the picture region. Fig. 4 (b) illustrates the second extension made to the left as a random

process. The path starts at the vertex with blue color, going through the alternating sequence of edge (gray) and vertex (red) and ends at the vertex of green color. Fig. 4 (b) and (c) are the intermediate results. Fig. 4 (d) is the final block spanning tree and the constructed Hamiltonian path.

Whenever one square unit of four vertices is added to the tree, four vertices and five edges are connected in the proper order and one edge at the extending position is deleted so as to make the path go through all the vertices which constitute the tree being constructed. When the tree finally becomes a block spanning tree, the path becomes the Hamiltonian path for the final block tree, which fills up all the vertices on the extended picture region. If the final block tree is a block spanning tree, the final path is the Hamiltonian path.

To construct the Hamiltonian path and the block spanning tree at once, a data structure of vector is useful to hold the positions of vertices and edges for the path being organized. The vector holds the sequence of vertices and edges in the order for which the vertices and edges appear on the path. Use of the vector provides an easy way to find a next block to be connected as well as the path part to which the next block path part is inserted. It makes it needless to traverse the tree while constructing the Hamiltonian path. With the vector, it is very easy to check if the vertices being connected are within the picture region and the connecting position is appropriate for the block. An experimentally organized vector is illustrated in section 6.

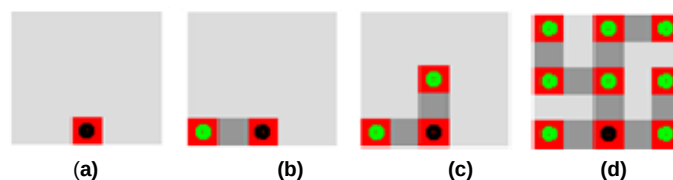


Fig. 3 How a tree grows and becomes a spanning tree

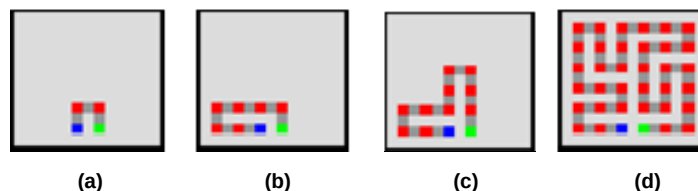


Fig. 4 Construction of block spanning tree with the block of four vertices and the Hamiltonian path

4.3 Experiments for Ordinary Binary Pictures

To construct a more practical maze, the binary picture of English letter 'B' was used to demonstrate the method of the block extension method. Fig. 5 (a) is the original binary picture and (b) is the size reduced picture. Fig. 6 (a) is the picture region of the 2-by-2 extended picture of Fig. 5 (b) and Fig. 6 (b) presents the region but with spaces of would-be edges and walls. Fig. 7 (a) is a maze constructed with start and goal colored by green and red, respectively. The start position is at the bottom of the picture 'B'. See all the positions (vertices) on Fig. 6 (b) are on the path in Fig. 7 (a). Fig. 7 (b) is another maze solution path with start and goal in the middle of the picture region. However, the solution for the maze can be easily found by right (or left) hand rule as pointed out by Okamoto and Uehara [6] and Hamada [8]. Fig. 8 (a) is the maze with random dead-end branches and (b) the one with walls thinned.

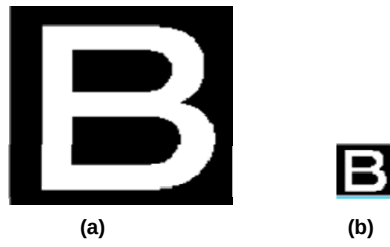


Fig. 5 Original binary picture (a) & (b) reduced by 4

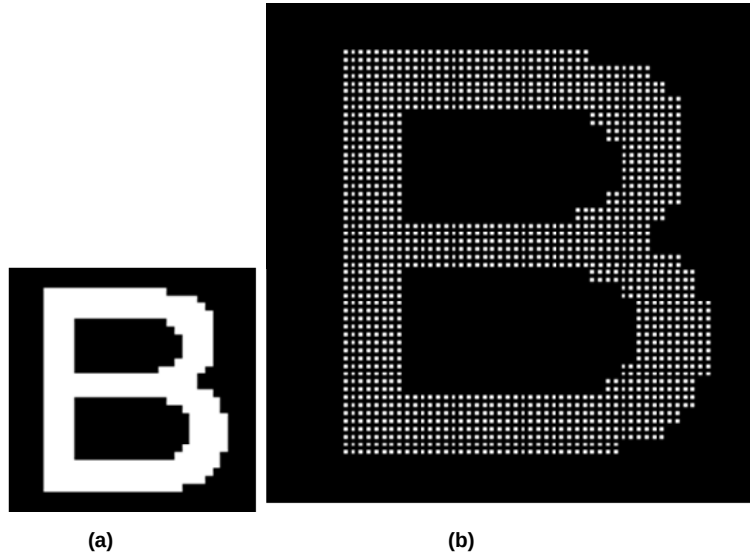


Fig. 6 (a) 2-by-2 extended picture and (b) would-be edges inserted picture

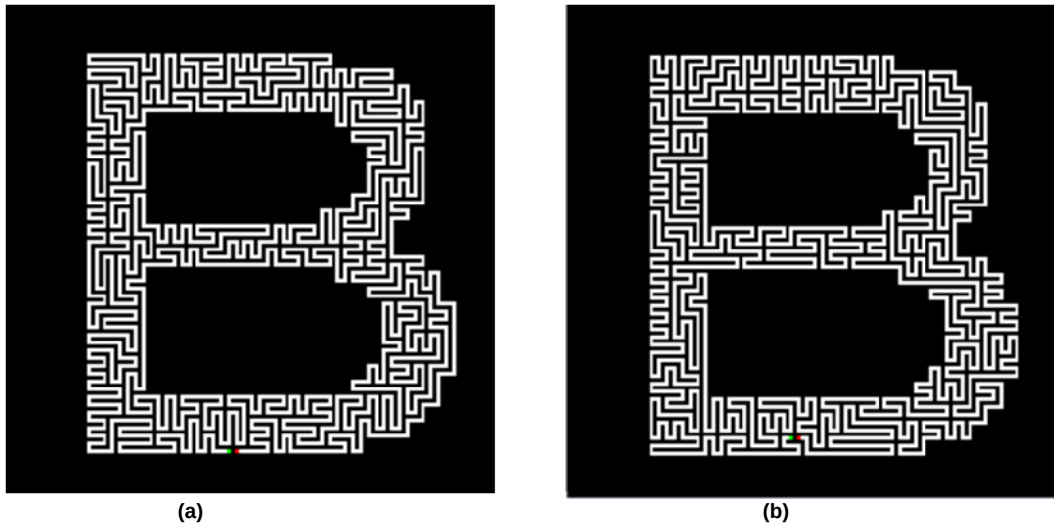
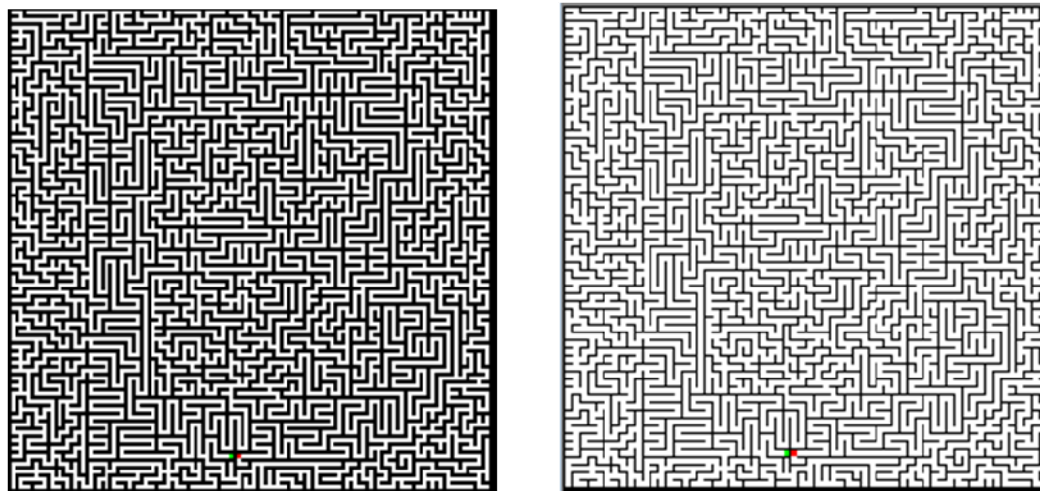


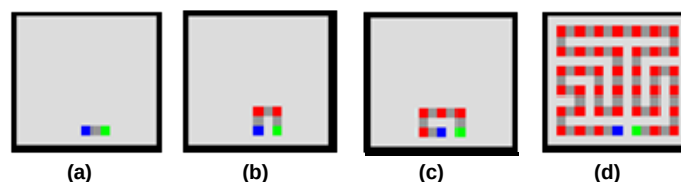
Fig. 7 (a) Block Hamiltonian path with start and end at the bottom, (b) block Hamiltonian path with start and goal in the middle of the picture.



(a) (b)
Fig. 8 Maze with random dead-end branches; (b) Same maze with thinned walls

5. PROPOSED GENERALIZED METHOD

Another block extension method was tried to demonstrate the picture maze generation using an ordinary binary picture instead of the 2-by-2 extended one. The block to extend the maze path includes only two vertices instead of four. As Fig. 9 (a) illustrates, it starts with two vertices (blue and green) and the edge between the two vertices. The first extension is upward as shown by Fig. 9 (b). Two vertices and three edges are also inserted into the vector in the proper order as done in the block method with four vertices of the previous section. The spanning tree completes as in Fig. 9 (d). This method is supposed to be more flexible because the block is smaller. This extension operation appears in the studies [9] by Ikeda and Hashimoto and [13] by Wong and Takahashi, but in different situations.



(a) (b) (c) (d)
Fig. 9 Construction of block spanning tree with the block of two vertices and the Hamiltonian path

5.1 Simple Application 1

A practical but simple sample picture was tried for a demonstration. Fig. 10 is the original picture of a circle; Fig. 11 is the reduced picture for which the maze is to be constructed. Fig. 12 is the initial path which consists of three line segments---the left and right line segments with some slope and the horizontal line segment inside the circle. The simplest way to make the initial path is to just draw a one line segment going through the circle as in the next subsection. But in this application, the initial path is made with three path segments. The entire path is from the entrance vertex (a red point at the extreme left) to the exit vertex (a green point at the extreme right). The path segment inside the circle is the base path from which a Hamiltonian path was tried to be constructed. The circle shown in Fig.12 is not essential but just shows the position of the initial path. Fig. 13 illustrates the beginning stage on which the block extensions proceeded to some extent. The path extension can be made at every position on the path inside the circle.

Fig. 14 is the maze solution path. The solution path should clearly depict the circle. Fig. 15 is the maze with random dead end branches and thinned walls. As seen in Fig. 14, there are a number of black vertices for which the solution path does not go through. The path failed to become a Hamiltonian path. Therefore, it should be called as a near Hamiltonian path. The complete avoidance of the black points is usually impossible. This is why the 2-by-2 extended picture was used in the study [6].

The entire initial path from the entrance to the exit can be the initial path from which the extension is to be made. However, the results should have essentially no change because the extension is allowed only inside the circle.

Straight path segments are often seen before the maze solution is obtained. However, those straight segments on the left and right sides of the circle are not seen on Fig. 15. So the hiding of those lines is considered to be well done. Horizontal or vertical lines were avoided as well as slopes about 45 degrees.



Fig. 10 Original circle



Fig. 11. Circle reduced by 2

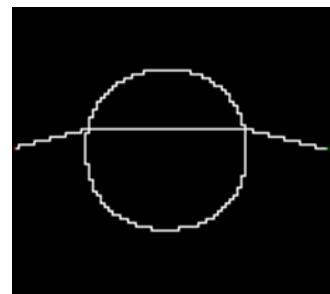


Fig. 12. Initial path

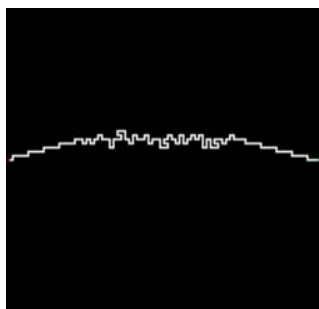


Fig. 13 Path extensions in some beginning stage

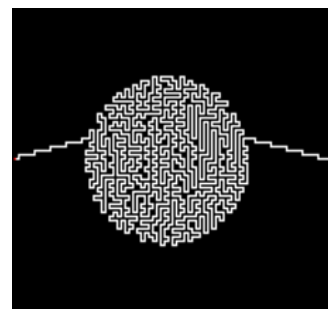


Fig. 14 Maze solution path completed for the circle

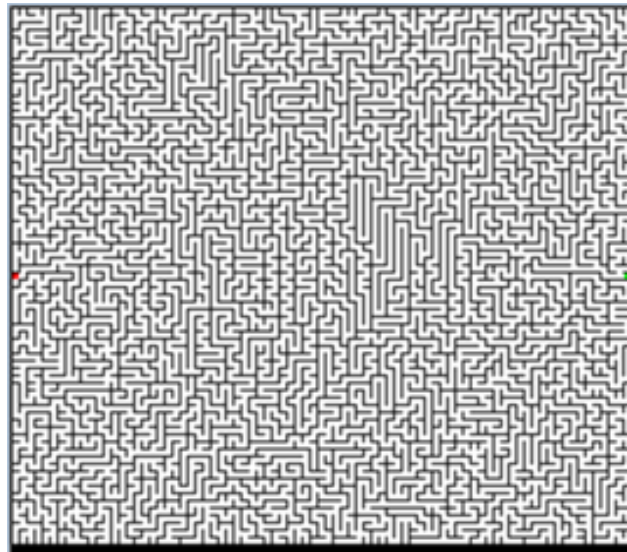


Fig. 15 Maze of a circle with thinned walls

5.2 Simple Application 2

A little more complex picture of 'AB' was tried for another demonstration. Fig. 16 is the picture of 'AB' which was already reduced by 6 from the original picture. The maze was to be constructed from this reduced picture. Fig. 17 illustrates the initial path which is just a straight path, the simplest, from the entrance vertex (a red point at the extreme left) to a boundary position of the exit vertex (a green point at the extreme right) passing through both of the foreground picture parts of 'AB'. There are two regions in which the path grows---'A' and 'B'. Those are the foreground parts of 'AB'. The contours of 'AB' in Fig. 17 are just to show where the initial path line goes through. The contour paths are not directly related with the path construction. Fig. 18 shows an early stage of the path extension. The extension occurs in both regions of 'A' and 'B'. This means that the entire path from the entrance to the exit was constructed at once.

Fig. 19 is the completed maze solution path. As the case of the circle in the previous subsection, a number of black dots are seen. These are the vertices not on the maze solution path. However, the picture 'AB' is clearly seen. Since the straight path lines are often easily seen before the solution is obtained [12], some zigzags were added to hide them, using the same path extension method, as shown in Fig. 20. After some dead end branches were added and walls were thinned, the final maze was finished as in Fig.21.

By the way, some more general procedures were suggested in the studies [9] and [13] for the initial path generation.



Fig. 16. Picture of 'AB' already reduced by 6



Fig. 17. Initial solution path for 'AB'

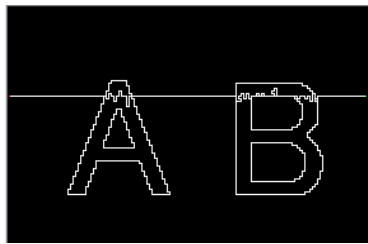


Fig. 18. Early stage solution of path extension for 'AB'



Fig. 19. Final solution path for 'AB'

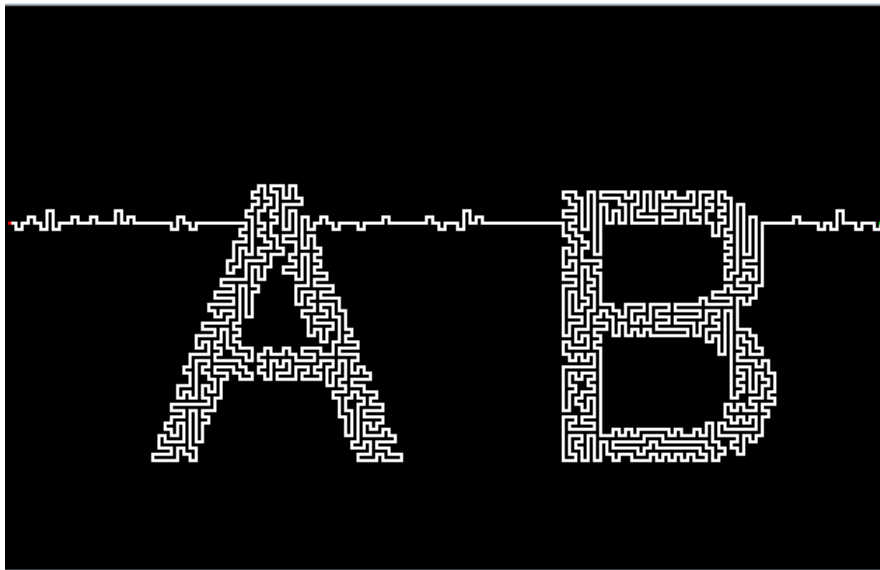


Fig. 20. Final solution path with zigzag for 'AB'

407
408

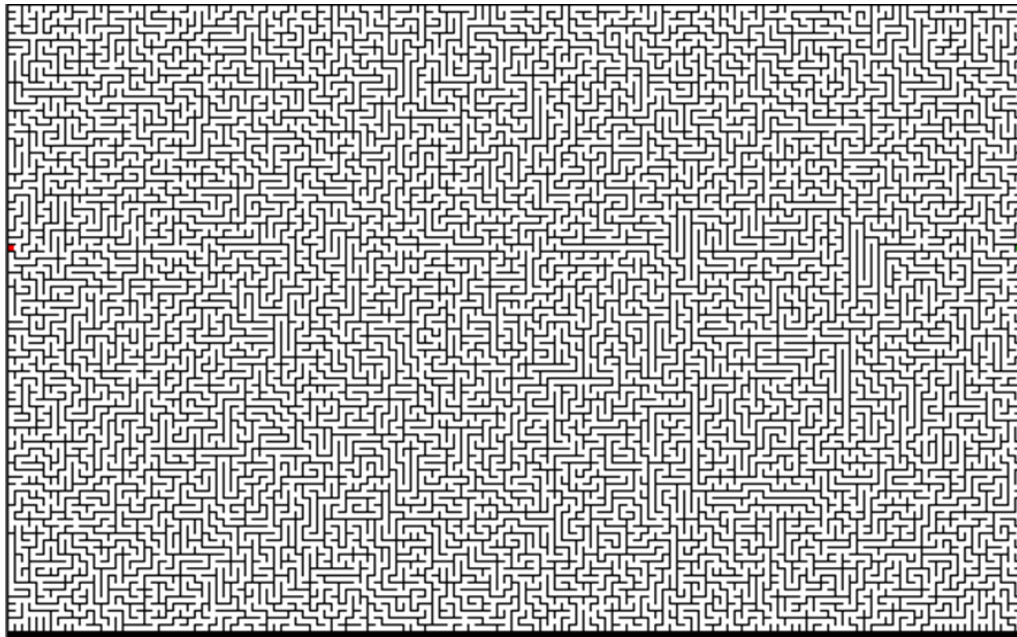


Fig. 21. Final maze for 'AB'

409
410
411
412

413 6. DISCUSSIONS

414 6.1 Algorithm of the Hamiltonian Path Construction and its Efficiency

415 In this subsection, more details are given about the construction of the block spanning tree
416 and the corresponding Hamiltonian path. Fig. 22 to 24 illustrate how the tree and the path
417 are developed on the same picture shape region as Fig. 4, that is, the 2-by-2 extended
418 picture region with 9 blocks. The path construction algorithm is given as:

- 419
- 420 1) Construct the path of the initial block in the array and the vector (Fig. 22 (a) and (b));
- 421 2) Find a proper position on the path to which a new block is connected;
- 422 3) Insert the four vertices and the five edges in the appropriate order into the array and the vector at
- 423 the proper position and delete one edge so that the vertices and edges together organize one path;
- 424 4) Repeat 2) to 3) until all blocks are connected.
- 425

426 Fig. 22 (a) is the initial path with the numbered vertices. The number, though not necessary
427 for the path construction, explains the order in which the vertices are inserted into the vector.
428 Fig. 22 (b) shows the vector in which the number and X and Y coordinate addresses on the
429 picture array are written. This illustrates that the path starts with the vertex of '4' and visiting
430 the vertices in the order '3', '2', and '1' which is the goal. X and Y addresses of vertex '4' to
431 '1' agree between Fig. 22 (a) and (b). Fig. 23 (a) and (b) show the situation where the
432 second block is connected. The vertex sequence in the vector is in Fig. 23 (b) which also
433 well corresponds with Fig. 23 (a). Fig. 24 is the final Hamiltonian path; the numbers attached
434 to vertices show the order in which they are inserted into the vector. The visiting order of the
435 vertex on the tree is illustrated on the path. The order is based on the right (or left) hand rule.

436

The proof of the Hamiltonian path construction algorithm is given as: 1) An ordinary spanning tree is always possible to construct. Then, a block spanning tree is always possible to construct on the 2-by-2 extended picture; 2) When a block is added to the tree being constructed, the edge between the two vertices is removed and the four vertices and 5 edges of the block in proper order (sequenced as edge and vertex alternately) are inserted into and between the two vertices. Then the constructed path goes through all the vertices of the tree being constructed; 3) When 2) is repeated until all blocks are inserted, the final path becomes a Hamiltonian path. Every time a block is added to the tree, four vertices of the block are inserted into the path without deleting a vertex. Therefore, when all blocks are processed, all the vertices are on the path, which is a Hamiltonian path.

The use of vector has at least two advantages as:

- 1) Avoidance of traveling around the spanning tree of section 3 and the article[6],
- 2) The region to be searched, when randomly finding an unconnected block and adding it to the block spanning tree, is very much limited. Only the vicinity of the growing tree is enough and it is given by the vector.

Table 1 shows the comparison about the number of branch extension trials between an ordinary spanning tree generation and that of Hamiltonian path construction using the block extension method and the vector. Six picture shapes were tested. They are given as the column name as 'S9', 'S16', 'S25', 'B83', 'B173', and 'B363'. The line of 'A' is for an ordinary spanning tree generation and the line of 'B' is for the Hamiltonian path generation using the block method and the vector. The foreground picture cell numbers are given on line 'S' and the frame array sizes are given on the line 'F'. The frame array is the area where the random branch extensions are tried and where the tree or the path grows. However, the random search area is given by the vector for the block extension method. The line of 'A/B' suggests that Hamiltonian path generation using the block extension method and the vector used much smaller number of position finding trials. This is especially true for 'B83', 'B173' and 'B363', where the numbers are about 3.34 times smaller for the block extension method. This implies that number of random trials gets smaller when the back ground is wider. It is reasonable that the use of vector is more advantageous for processing the picture 'B' which has a wide background area. The picture 'B' has wider background regions compared to the square shapes--'S9', 'S16' and 'S25'--which have no background area for which the block method need about 1.36 times less trials. Since a practical picture has usually a wider background area, the method using the vector is advantageous. By the way, the spanning tree for 'B363' is given in Fig.25.

However, how much overhead exist for the use of the vector is not known at this point. The program execution on the personal computer (MAC PRO with 2x2 2.8 GHz Quad-Core Intel Xeon; purchased Feb. 2008) can be finished in a moment for obtaining the maze of Fig. 8 (b), Fig.15 and Fig.21. The additional merit of using the block extension is that it can be extended further to more general method with two vertices.



(a) (b)
Fig. 22 Initial block of four vertices

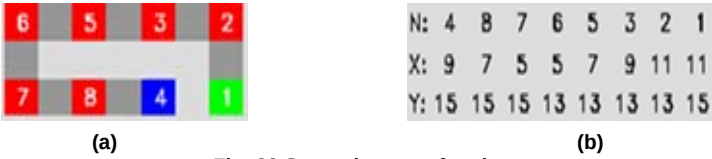


Fig. 23 Second stage of path



Fig. 24 Final Hamiltonian path

Table 1 Comparison of number of random position finding trials for branch extension between A and B
A: A spanning tree construction on a binary picture. In computing the numbers on A, a pair of random numbers x and y are counted as one.
B: A Hamiltonian path construction on the corresponding 2-by-2 extended picture. All numbers given on lines of A and B are averages of 10 experiments.
S9: 3*3 square binary picture like Fig.3 (d): the entire part is the foreground.
S16: 4*4 square binary picture: the entire part is the foreground.
S25: 5*5 square binary picture: the entire part is the foreground.
B83: The binary picture of 'B' with 49 vertices on the foreground part in x*y picture frame.
B173: The binary picture of 'B' with 173 vertices on the foreground part in 19*21 picture frame.
B363: The binary picture of 'B' with 363 vertices on the foreground part in 31*31 picture frame.

S: Shape with Foreground Vertex Numbers	S9	S16	S25	B83	B173	B363
F: Frame Size	3*3	4*4	5*5	15*15	21*19	31*31
A: Ordinary Spanning Tree Construction	79	177	347	11405	25372	85391
B: Hamiltonian Path Construction by Block Method	55	133	267	3554	8316	22683
A/B	1.44	1.33	1.30	3.21	3.05	3.76



Fig. 25 An ordinary spanning tree for 'B363'

533 **6.2 Near Hamiltonian Path by Block Extension with Two Vertices**

534 A Hamiltonian path can always be constructed on the 2-by-2 extended picture if it is
 535 connected. However, it could not usually be done on an ordinary picture, non-extended one.
 536 There are some vertices left, not inserted on the path, which are shown as black dots within
 537 the circle in Fig. 14. Some could be removed as [9]; but it is not usually possible to delete all
 538 the dots. That is, it is not usually possible to construct a Hamiltonian path for an arbitrary
 539 connected shape. This is why the 2-by-2 extended picture was used in subsection 4.2. Even
 540 though there are some vertices left, not included on the path, the solution path for the maze
 541 clearly depicts the picture.

542
 543 Since the block extension with two vertices uses the same method with the vector of the
 544 previous subsection, the algorithm is very simple and fast. The method [9] employed
 545 Simulated Annealing and obtained similar results, possibly better with fewer black dots, but
 546 reportedly consumed more time, about 30 seconds on the picture array of 160x120 vertices.

547 **6.3 Right (Left) Hand Rule**

548 The Hamiltonian path constructed in the subsections 4.2 and 4.3 can be traversed by the
 549 right (left) hand rule. It is because of the fact that the path was constructed from one initial
 550 block. Once the extension is started and as long as the subsequent extension is based on
 551 the initial extension, the constructed path follows the same hand rule. However, if the initial
 552 block positions are different and the initial extension is to the different direction, the situation
 553 is different. For the case of Fig. 12, the initial block position can be anywhere on the straight
 554 line in the circle as demonstrated in Fig. 13. The extension can be to the top or to the bottom
 555 at first. When the first extension on the line is to the top, the subsequent path extended from
 556 the first one will be of the same hand rule. When it is to the bottom, the subsequent path is of
 557 the left (right) hand rule. As shown in Fig. 13, there are a number of initial extensions in both
 558 directions. So the two kinds of path are mixed in the solution path in Fig. 14, the maze would
 559 become harder to solve. For the case of 'AB', the initial block extension can be upward or
 560 downward on the picture regions of 'AB' as shown in Fig. 17 and 18. Therefore the solution
 561 is not simple.
 562

563 **6.4 Path Lines Seen before the Maze Is Seemingly Solved**

564 Some maze solution path segments were often seen before the maze was solved in the
 565 paper [12]. However, this problem is not seen noticeably in the results of this study. The
 566 horizontal, vertical or near 45 degree straight lines were often seen before the solution is
 567 made. These horizontal straight lines are avoided with some zigzags as in Fig. 20. The
 568 straight lines were drawn with some slopes but not around 45 degree or near vertical or
 569 horizontal. Therefore much improvement has been accomplished in this study as long as the
 570 pre-seen solution path lines are concerned.

571 **7. CONCLUSION**

572 A simple and unified method for constructing picture mazes for a 2-by-2 extended picture as
 573 well as non-extended one was presented. It was proved to be easily applicable to a variety
 574 of binary pictures. The algorithm is based on the construction of a vector to hold the
 575 Hamiltonian path or near Hamiltonian path. Experiments and discussions demonstrated the
 576 method is simple and efficient.

577

578 **REFERENCES**

579

580 [1] "Maze generation algorithm," http://en.wikipedia.org/wiki/Maze_generation_algorithm,
 581 Accessed 2013/01/09

- 582 [2] T. Hosokawa, "Maze," <http://aanda.system.to/> Accessed 2013/01/25
- 583 [3] J. Xu and C. S. Kaplan, "Image-Guided Maze Construction," *ACM Transactions on Graphics*, 26,
- 584 3, Article 29, pp.1-9, 2007
- 585 [4] J. Xu and C. S. Kaplan, "Vortex maze construction," *Journal of Mathematics and the Arts*, 1, (1),
- 586 pp.7-20, 2007
- 587 [5] Conceptis puzzles, "Maze-a-PiX," <http://www.conceptispuzzles.com/>, accessed 2013/01/09
- 588 [6] Y. Okamoto and R. Uehara, "How to make a picturesque maze," *Proc. Canadian Conf. on*
- 589 *Computational Geometry*, pp.137-140 , 2009
- 590 [7] H. Hamada, <http://www.lab2.kuis.kyoto-u.ac.jp/~itohiro/Games/100301/100301-11.pdf> ,
- 591 accessed 20013/01/12
- 592 [8] K. Hamada, "A Picturesque Maze Generation Algorithm with Any Given Endpoints," *Journal of*
- 593 *Information Processing*, Vol. 21, No. 3, pp.393-397, 2013
- 594 [9] K. Ikeda and J. Hashimoto, "Stochastic Optimization for Picture Maze Generation," *Jouhou Shori*
- 595 *Gakkai Ronbunshi*, Vol. 53, No.6, pp.1625-2634, 2012, in Japanese
- 596 [10] T. Kurokawa, K. Mori, and T. Mizuno, "Automatic construction of picture maze by repeated
- 597 contour connection," *Thailand-Japan Joint Conference on Computational Geometry and Graphs*,
- 598 pp.5-6, 2012
- 599 [11] T. Kurokawa, "Picture maze generation and its relation to graphs," 2013 SHIKOKU-SECTION
- 600 JOINT CONVENTION RECORD OF THE INSTITUTE OF ELECTRICAL AND RELATED
- 601 ENGINEERS, p.231, 2013
- 602 [12] T. Kurokawa, "Picture Maze Generation by Repeated Contour Connection and Graph Structure
- 603 of Maze," *Computer Science and Engineering*, Vol.3, No.3, pp.76-83, 2013.12, Scientific &
- 604 Academic Publishing, USA , ISSN:2163-1484
- 605 <http://article.sapub.org/10.5923.j.computer.20130303.04.html> Accessed 2014/1/20
- 606 [13] F. J. Wong and S. Takahashi, "Flow-based Automatic Generation of Hybrid Picture Mazes,"
- 607 *Computer Graphics Forum*, 28, (7), pp.1975-1984, Wiley-Blackwell, 2009
- 608 [14] S. Ishida, "Algorithm of automatic maze generation,"
- 609 <http://www5d.biglobe.ne.jp/%257estssk/maze/make.html>, (Japanese), Accessed 2013/10/8
- 610 [15] "How to Build a Maze," <http://www.mazeworks.com/mazegen/mazetut/index.htm>, Accessed
- 611 2013/1/26
- 612 [16] <http://www.microsoft.com/ja-jp/dev/express/default.aspx>, Accessed 2012/10/15
- 613 [17] <http://jaist.dl.sourceforge.net/project/opencvlibrary/opencv-win/2.1/> Accessed Oct., 2011