

Construction of Picture Maze along Set of Image Dot Vertices

Tomio Kurokawa*

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

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

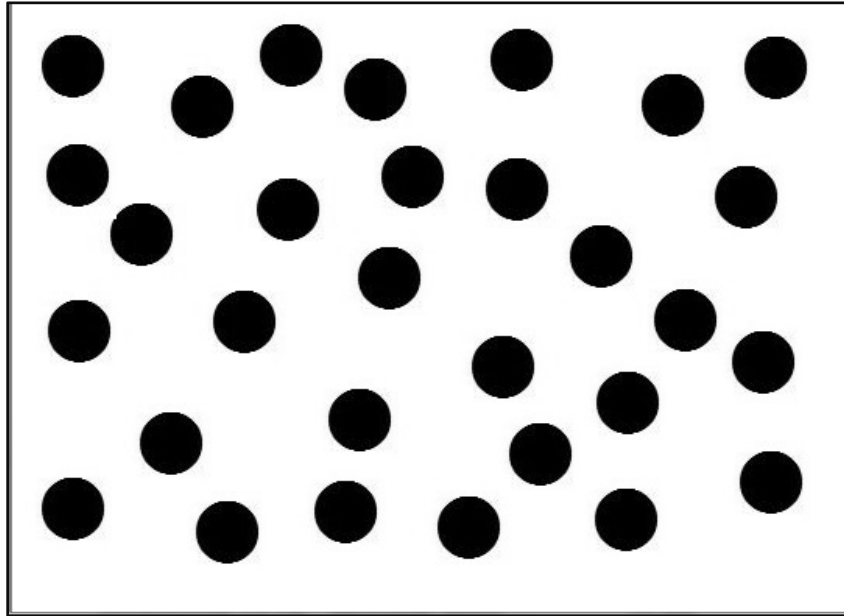
1. INTRODUCTION

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. 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 is designated as the start and the goal, then the path between the two is the solution.

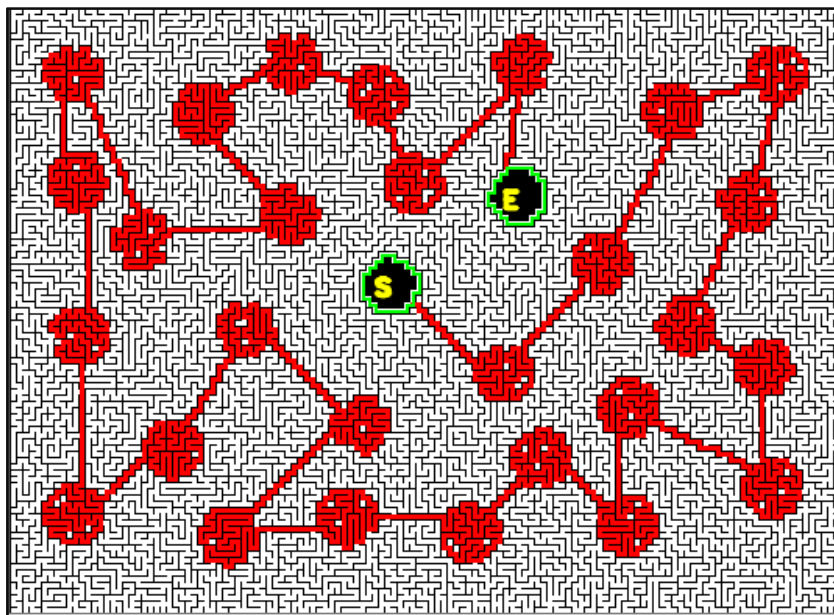
A maze constructed in this study is based on the maze structure above. But it is generated based on an original binary picture. Employing techniques of image processing, graph theoretical concepts, and following the constraints of the maze structure, a kind of complex maze (picture maze) is generated. 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

37 with the solution path (red) painted on it. If the dots have some shapes, then some pictures
38 could appear accordingly.
39
40
41
42



63 Fig. 1 Original picture of image dots; number of dots=30
64
65



86 Fig. 2 Constructed maze of image dots with solution path
87
88
89

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

$$cycle_j = \{ (x(j,i), y(j,i)) \mid 1 \leq i \leq n_j \} \quad (1)$$

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

$$CYCLES = \{ cycle_j \mid 1 \leq j \leq n \} \quad (2)$$

Each cycle is considered as a vertex of graph:

$$cycle_j = v_j \quad (3)$$

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 mean 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 $cycle_n cycle_1$.

$$P = cycle_1 cycle_2 \dots cycle_n \quad (4)$$

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

$$P = v_1 v_2 \dots v_n \quad (5)$$

DEFINITIONS:

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

$$d(j, k) = \min_{1 \leq p \leq n_j \text{ \& } 1 \leq q \leq n_k} \sqrt{(x(j, p) - x(k, q))^2 + (y(j, p) - y(k, q))^2} \quad (6)$$

$P(j)$ is the j th 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 $d(P(a), q) + d(q, P(b))$ 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

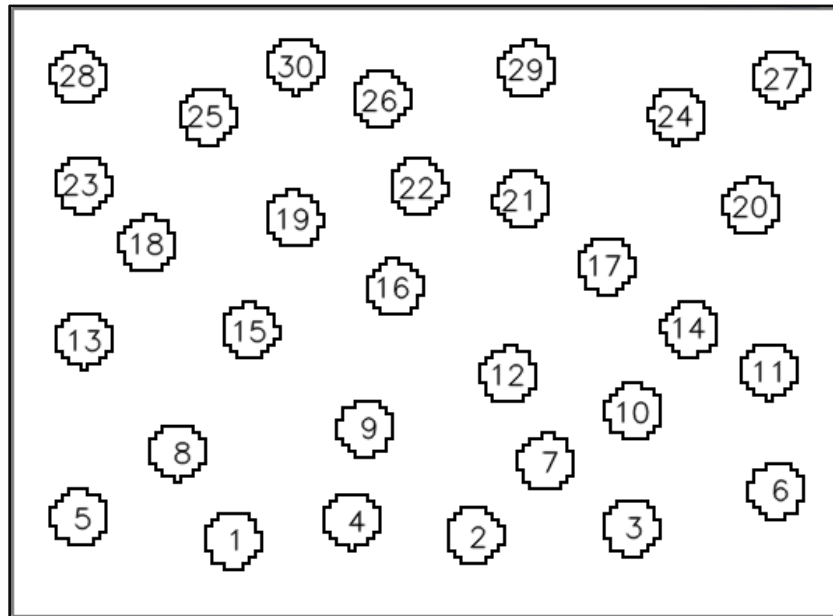


Fig.3 Dot contour cycles

283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333

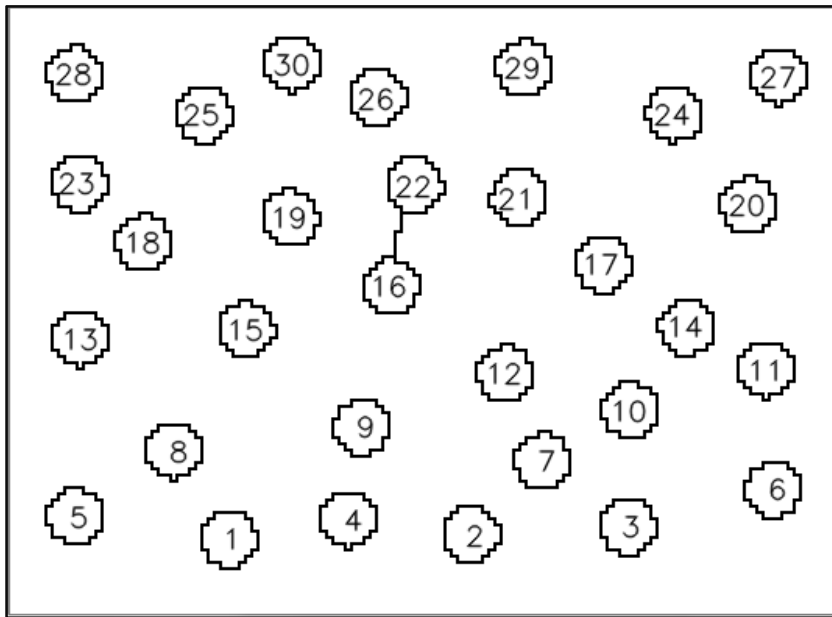


Fig.4 First connection

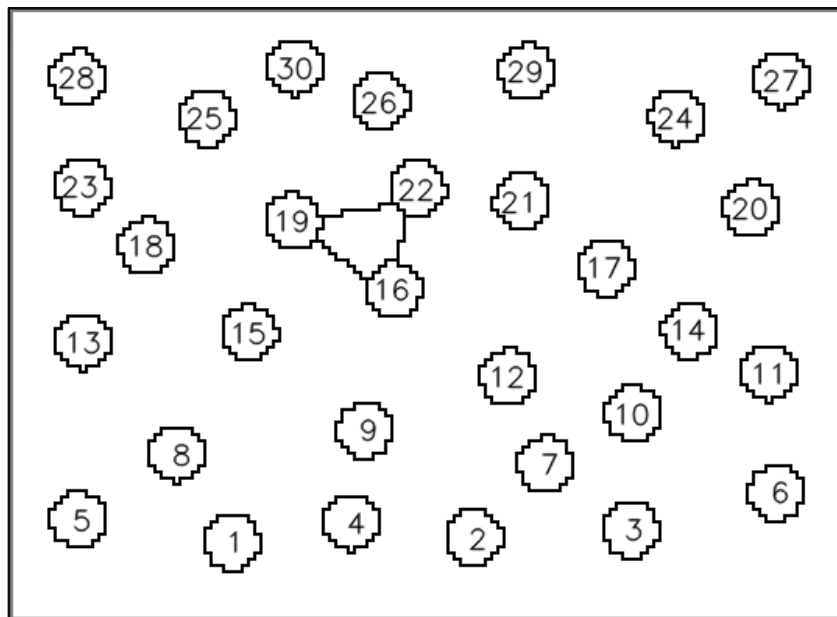


Fig. 5 Three dot cycle closed path

334
 335
 336
 337
 338
 339
 340
 341
 342
 343
 344
 345
 346
 347
 348
 349
 350
 351
 352
 353
 354
 355
 356
 357
 358
 359
 360
 361
 362
 363
 364
 365
 366
 367
 368
 369
 370
 371
 372
 373
 374
 375
 376
 377
 378
 379
 380
 381
 382
 383
 384
 385
 386

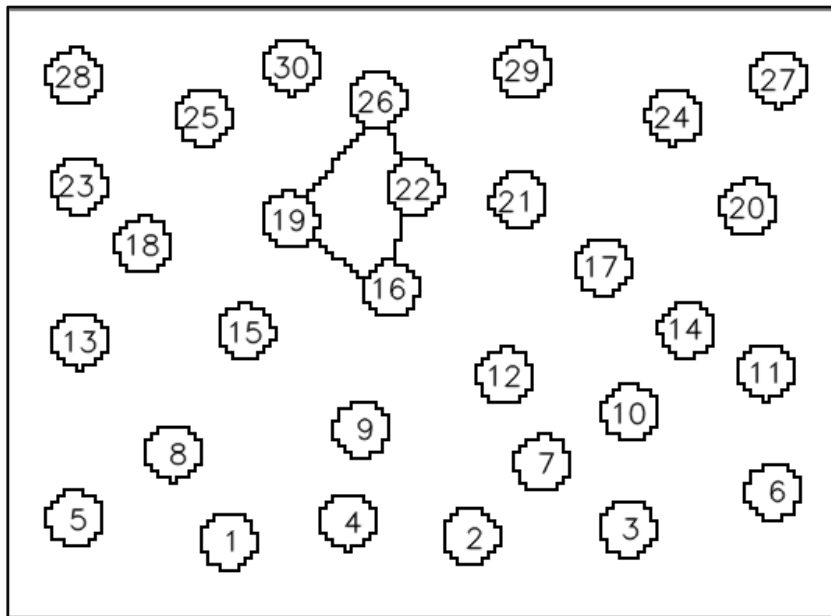


Fig.6 No. 26 is inserted in the closed path

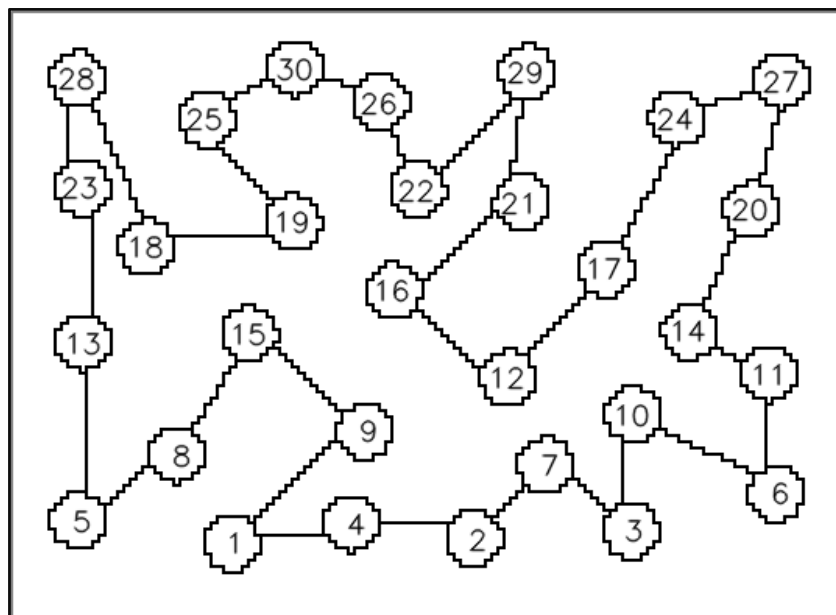


Fig. 7 Closed path completed

5 OBTAINING COMPLETE MAZE AND DISCUSSIONS

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 completed. 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.

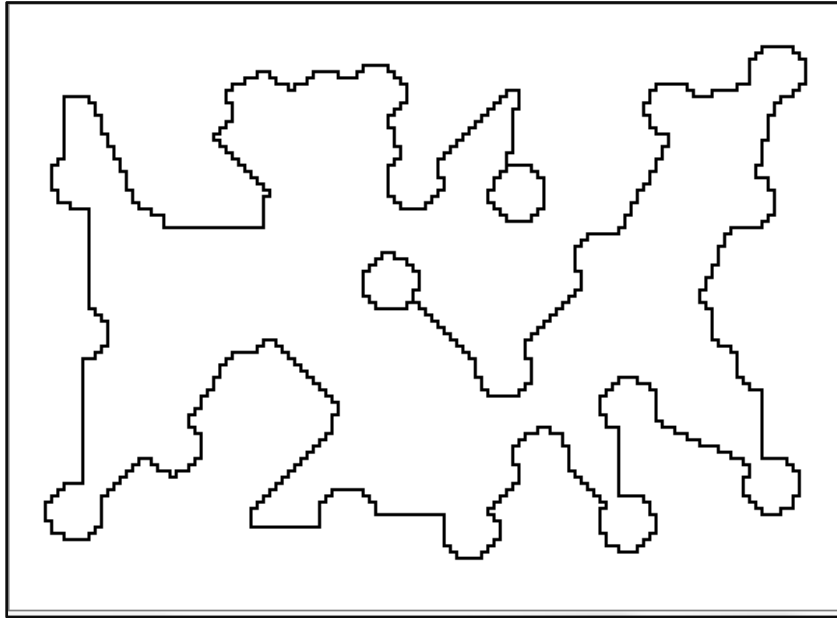
5.2 CONSTRUCTION OF PICTURE MAZE

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

439 image dots, many different path constructions are not possible as long as the start and end
440 are by a pair of the dots.
441
442



443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466 Fig.8 Starting base line of solution path
467
468
469

470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520

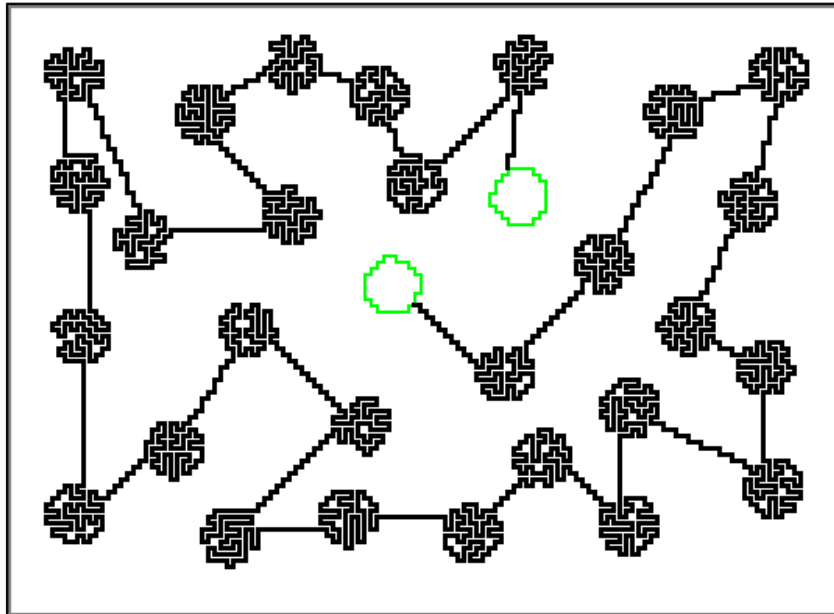


Fig.9 Maze solution path completed

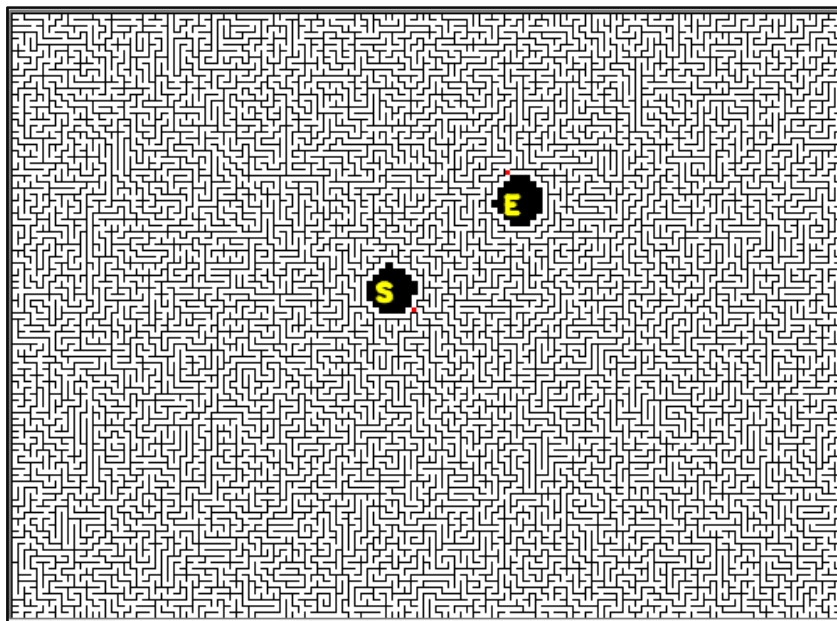


Fig.10 Final maze completed to be presented to players

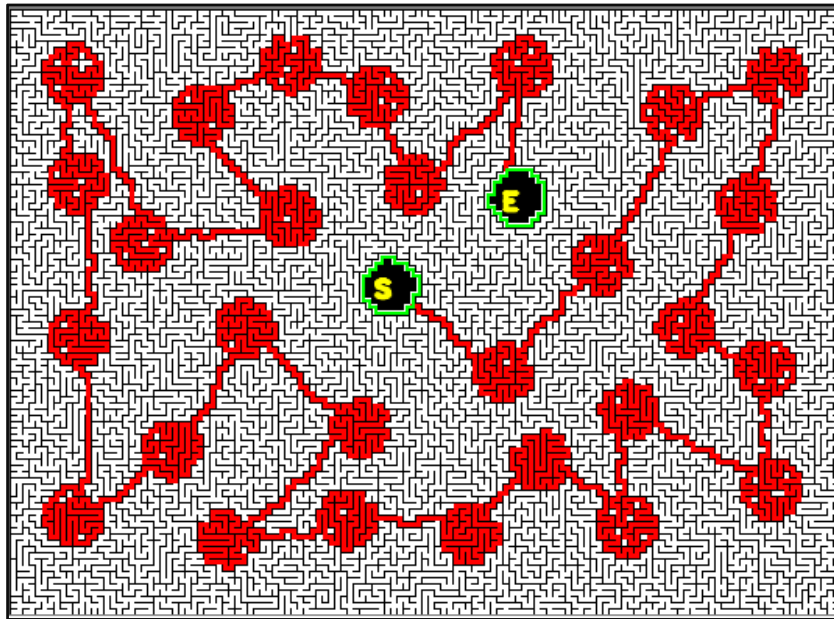


Fig.11 Final maze but with curved solution path lines

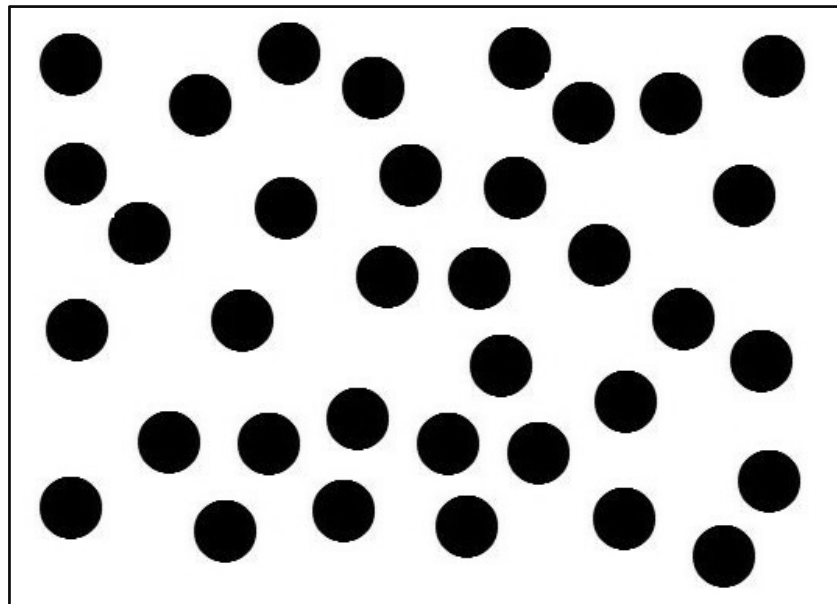


Fig. 12 Original picture of image dots; number of dots=35

574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626

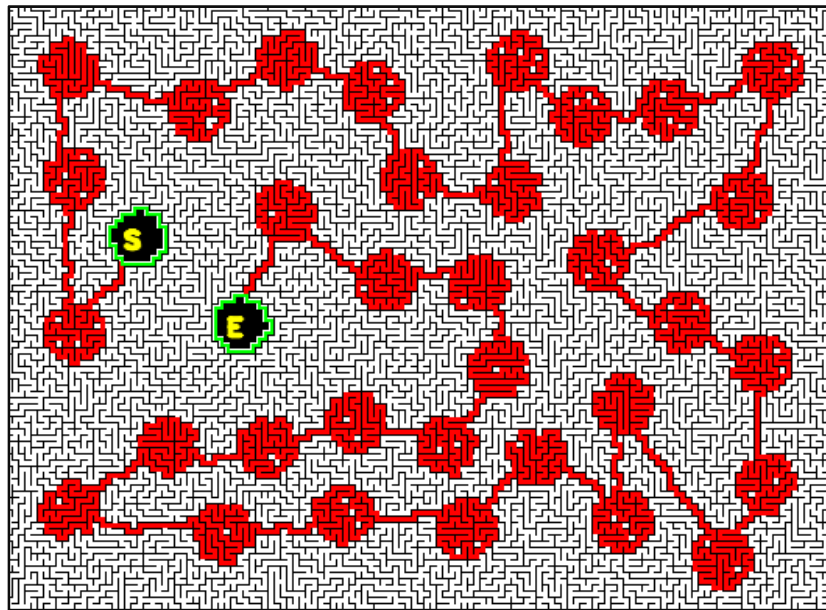


Fig. 13 Final maze with curved solution path lines

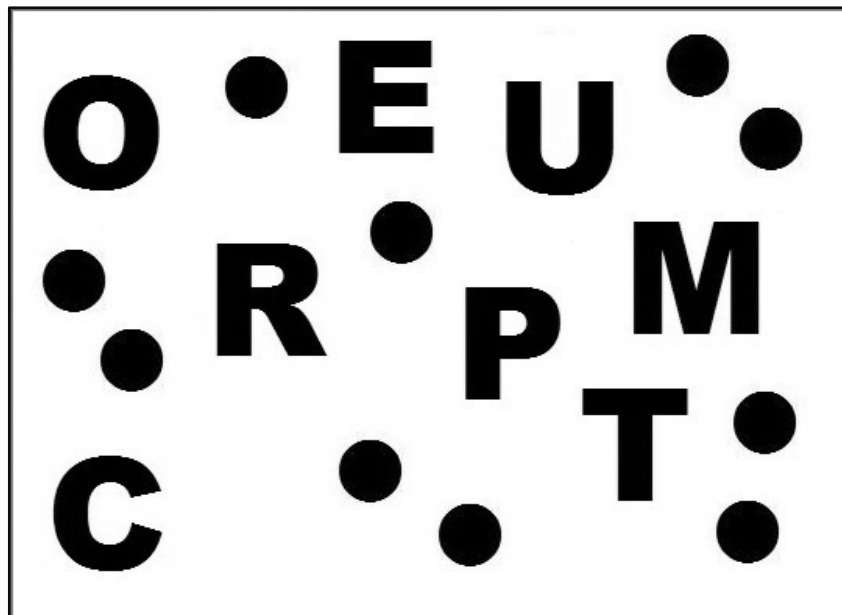


Fig. 14 Original binary picture with the alphabets of "COMPUTER"

627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679

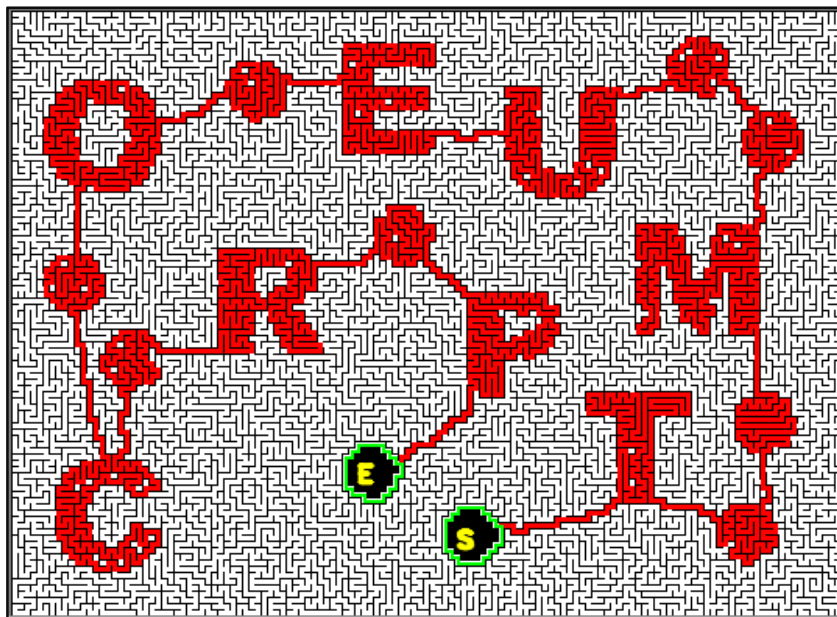


Fig. 15 Complete quiz maze with solution path to find "COMPUTER"



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 $d(P(a), q) + d(q, P(b))$ 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:

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

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

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

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

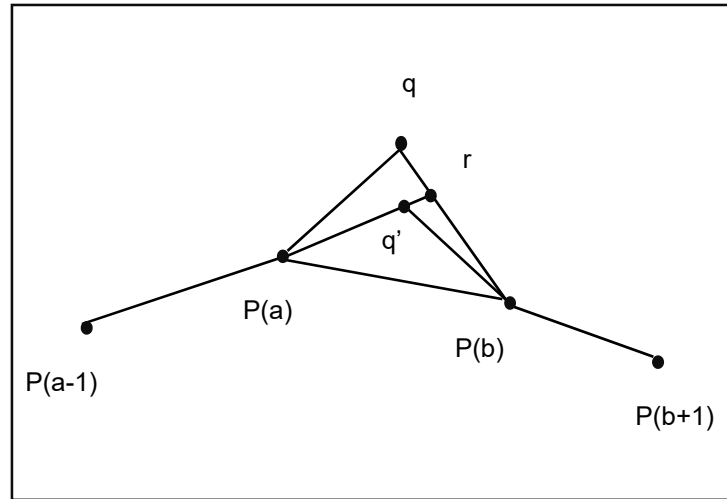


Fig. 18 Triangle $(P(a), q, P(b))$. b is the next item number of a in P which is circular.

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.

786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825

REFERENCES

- [1] T. Hosokawa, "Maze," <http://aanda.system.to/> Accessed 2016/05/6
- [2] "Maze generation algorithm," http://en.wikipedia.org/wiki/Maze_generation_algorithm, Accessed 2016/05/6
- [3] J. Xu and C. S. Kaplan, "Image-Guided Maze Construction," ACM Transactions on Graphics, 26, 3, Article 29, pp.1-9, 2007
- [4] J. Xu and C. S. Kaplan, "Vortex maze construction," Journal of Mathematics and the Arts, 1, (1), pp.7-20, 2007
- [5] Conceptis puzzles, "Maze-a-PiX," <http://www.conceptispuzzles.com/>, Accessed 2016/05/6
- [6] Y. Okamoto and R. Uehara, "How to make a picturesque maze," Proc. Canadian Conf. on Computational Geometry, pp.137-140, 2009
- [7] K. Hamada, "A Picturesque Maze Generation Algorithm with Any Given Endpoints," Journal of Information Processing, Vol. 21, No. 3, pp.393-397, 2013
- [8] T. Kurokawa, "Picture Maze Generation by Successive Insertion of Path Segment," British Journal of Mathematics and Computer Science, Vol.4, Issue.24, pp.3444-3463, 2014.12, Science Domain international, ISSN:2231-0851, <http://www.sciencedomain.org/issue.php?iid=699&id=6>
- [9] K. Ikeda and J. Hashimoto, "Stochastic Optimization for Picture Maze Generation," Jouhou Shori Gakkai Ronbunshi, Vol. 53, No.6, pp.1625-2634, 2012, in Japanese, <https://dSPACE.jaist.ac.jp/dspace/bitstream/10119/10912/1/18003.pdf>, Accessed 2016/5/6
- [10] T. Kurokawa, K. Mori, and T. Mizuno, "Automatic construction of picture maze by repeated contour connection," Thailand-Japan Joint Conference on Computational Geometry and Graphs, pp.5-6, 2012
- [11] T. Kurokawa, "Picture maze generation and its relation to graphs," 2013 SHIKOKU-SECTION JOINT CONVENTION RECORD OF THE INSTITUTE OF ELECTRICAL AND RELATED ENGINEERS, p.231, 2013
- [12] T. Kurokawa, "Picture Maze Generation by Repeated Contour Connection and Graph Structure of Maze," Computer Science and Engineering, Vol.3, No.3, pp.76-83, 2013.12, Scientific & Academic Publishing, USA , ISSN:2163-1484 <http://article.sapub.org/10.5923.j.computer.20130303.04.html> Accessed 2016/05/6
- [13] F. J. Wong and S. Takahashi, "Flow-based Automatic Generation of Hybrid Picture Mazes," Computer Graphics Forum, 28, (7), pp.1975-1984, Wiley-Blackwell, 2009
- [14] T. Kurokawa, "Maze Construction by Using Characteristics of Eulerian Graphs," British Journal of Mathematics & Computer Science, Vol.9, Issue.6, pp.453-472, 2015.6, SCIENCE DOMAIN international, UK, ISSN: 2231-0851, <http://sciencedomain.org/issue/1147>