3D L-System Pathfinding with Genetic Algorithm

Djuned Fernando Djusdek
Dept. of Computer Science and Engineering
Toyohashi University of Technology
Toyohashi, JAPAN
djuned@sys.cs.tut.ac.jp

Yoshiteru Ishida
Dept. of Computer Science and Engineering
Toyohashi University of Technology
Toyohashi, JAPAN
ishida@cs.tut.ac.jp

ABSTRACT

L-System and Extended L-System for pathfinding is an approach to exploring unexplored land. However, this covers only a few cases. In this study, we propose general solutions for 2 and 3-dimensional exploration. We use the Genetic Algorithm (GA) as an L-Systems' grammar modifier that combines with a spherical coordinate system to calculate the next growth position. By using GA, we can quickly develop 3D exploration for flying agents by adding new genes. In this paper, the experiments were done separately between walking agents and flying agents' exploration. The experimental results are validated and performed in the simulation system. The performance is verified by calculating the percentage of the coverage area, the execution time and the shortest path distance. In this research, by using GA the distance of the explored path is reducing — however, it takes longer execution time.

CCS Concepts

• Theory of computation → Formal languages and automata theory → Grammars and context-free languages.

Keywords

L-System; grammar; Genetic Algorithm; pathfinding; spherical coordinate system.

1. INTRODUCTION

L-System and Extended L-System for pathfinding is a kind of technique for exploring a map and creating a path simultaneously. Recently, this work focused on 2D maps (Dimension) [1] and some also in 3D, but only for walking agents [2]. The technique used is simple, using just deterministic grammar and some rules. This work focuses on the scope of the exploration area and does not yet have quantitative measurements. In [1], they are using a stochastic approach for modified Extended L-System (growing and shrinking). However, in [2] while only use a deterministic approach and modifier rule, called *flipping approach*.

In this study, we tried to improve previous work [2] using the GA [3]. GA is a kind of approach for solving the NP-problems (Non-deterministic Polynomial Time) such as TSP (Travelling Salesman Problem). This algorithm only produces a solution that is close to optimal with some parameters and rules. GA is an iterative method to solve one problem. This condition creates another problem, "how much do we need to iterate." So, we try to settle using acceptable rules for our work.

GA is used to replace the *flipping approach* that was introduced in [2] and its effect in generality pattern. Besides, we expanded the L-System with GA to complete 3D exploration for flying agents. Flying agents are an agent that is not required to touch the surface, e.g., drone. On the other hand, walking agents are an agent that needs to contact the surface, e.g., human.

The rest of this paper is structured as follows. Section 2 describes the previous research which relates to L-System and Extended L-System for pathfinding. Section 3 presents the method we propose with which the simulation and experimental results are provided in Section 4 and 5 — finally, we offer our conclusions in Section 6.

ICISS 2019, March 16–19, 2019, Tokyo, Japan
© 2019 Association for Computing Machinery.
ACM ISBN 978-1-4503-6103-3/19/03...$15.00
https://doi.org/10.1145/3322645.3322684

2. RELATED WORK

Referring to [2], in this study we expanded the L-System rules by combining with the GA. Some previous research has done this. Most studies focused on representing more natural plants in the term of evolution studies as described in [4].

Recently, GA was used with L-System, e.g. [5]. In this research Bernard et al. introduced the Plant Model Interface Tool (PMIT) that infers deterministic context-free L-System from an initial sequence of strings generated by the system using a genetic algorithm. PMIT can deduct more complex systems than existing approaches. Indeed, while existing methods are limited to L-System with a total sum of 20 combined symbols in the productions, PMIT can infer almost all L-Systems tested where the total amount is 140 symbols.

Then, with L-System and Extended L-System for Pathfinding, several studies have been conducted, such as [1] and [2]. In [1], they apply generative rules (growing dynamics) and death rules (shrinking dynamics), where they use a stochastic approach to solve them in 2D maps. Then, [2] only focuses on generative rules that develop deterministically for 3D maps. In [2], *flip* and *keep* actions as the L-Systems' grammar modifier are proposed.

Related to [2]. In this study, we propose a general rule for decoding the *flip* and *keep* actions. Then, we expand it to support a *top-down* growth direction as well.

3. METHOD

In this study, we use GA as L-Systems' grammar modifier. Related to [2], here the *flip* and *keep* actions are defined as a gene of a chromosome. This chromosome contains the direction information of growth, i.e., θ value, represent as L-Systems' grammar, "plus sign" (+) and "minus sign" (-) for *left-right* direction (in 2D). Also, "and sign" (&) and "caret sign" (^) for *top-down* direction (added for 3D). Then use the spherical coordinates system (Eq. 1) to calculate the growing position, where θ is the polar angle and φ is the azimuthal angle [6].

$$x = r \sin\theta \cos\varphi$$
$$y = r \sin\theta \sin\varphi \qquad (1)$$
$$z = r \cos\theta$$

On Fig. 1 shows an illustration of the growing action for chromosomes: +-&^. The location of the next path will be calculated from the gene value. The first gene is used for first growth and so on. When the chromosome reaches the last gene, the process will repeat from the first gene, until all areas that may be visited are explored.

Steps of GA used are selection, crossover, and mutation. In the selection's step, will select two unique chromosomes using Pseudo RNG (Random Number Generator) using Mersenne Twister's algorithm [7], with the specific seed (fixed seed). Then, in the crossover's step, these two chromosomes will join or mix with portions depending on the selected parent fitness score, where the first portion of the chromosome comes from the first chromosome chosen and the second portion of the chromosome comes from the second one. Lastly, in the mutation's step, all the genes within the chromosome will mutate.

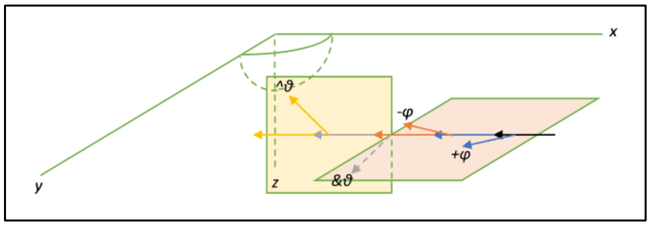

Figure 1. Illustration of Chromosome: +-&^.

Algorithm: update-fitness (found-path, current-steps)

```
1.  initiate min-steps with MAX
2.  initiate higher-fitness with 0
3.  if found-path = true
4.      s ← (fitness / 2) * (min-steps / current-steps)
5.      increase fitness with s
6.      increase parent's fitness by half of s
7.      if current-steps < min-steps
8.          min-steps ← current-steps
9.          if higher-fitness > s
10.             f ← (higher-fitness + s) / 2
11.             increase fitness with f
12.             increase parent's fitness by half of f
13.     if higher-fitness < s
14.         higher-fitness ← s
15. else
16.     fitness ← 0
17.     decrease parent's fitness by dividing by 2
18. return
```

Figure 2. The Pseudocode of Fitness Function.

GA also requires a fitness function to control the number of iterations. In this study, we propose rules for the calculation of fitness score. The fitness function uses a rating of 0.8 for minimum stopping iterations (termination rule). That means when the fitness score in the current iteration is higher than 0.8 iterations it will stop, and the result will be selected as the solution.

However, using fitness score alone does not ensure that the results will be better than *flipping approach* [2]. One solution is to compare with the shortest path result from all iterations (global shortest path), then use them to update the fitness score and control the iteration. This algorithm is shown in Fig. 2, where the fitness score is calculated from the shortest path distance.

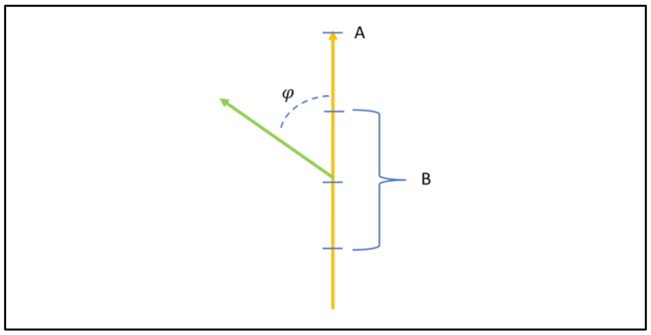

Figure 3. The Branching Method.

Besides, GA will be extended using a stochastic approach; then we called *stochastic GA*. This approach adds a random value next to the gene. This random value will affect the degree and the position of the branching from the source. See Fig. 3, ω is the branching degree (ω representing θ and φ), A and B are the next positions of the source path, where B has three possible locations to be determined using random values.

The rules used are:

1. Random value = [0, 0.333]
   a. ω 15° closer to A; and
   b. B is ¼ from the root to A.
2. Random value = [0.334, 0.666]
   a. ω will use defined degree; and
   b. B is ½ from the root to A.
3. Random value = [0.667, 1]
   a. ω 15° further than A; and
   b. B is ¾ from the root to A.

From those rules, point *a* will be called *probabilistic degree* and the combination of points *a* and *b* will be called *probabilistic degree and branching position*.

In this study, we divide the experiment for walking agents and flying agents. Because in flying agents there exists not only *left-right* but also *top-down* growth. So, this has more possibilities than walking agents (only *left-right* growth direction). All experiments will be done in the simulation system.

4. SIMULATION

In this study, we extend the simulation system from [2] to support this research. The specifications for this simulation system are described:

4.1 Hardware Specifications

In this research, the simulation built to focus on:

- MacBook Pro (13-inch, 2017), with specifications:
  - Processor: 3.1 GHz Intel Core i5-7267U
  - Memory: 8 GB 2133 MHz LPDDR3
  - Graphics: Intel Iris Plus 650 1536 MB

4.2 Software Specifications

This simulation system builds on:

- MacOS 10.14 Mojave;
- Qt Creator 4.4.1, based on Qt 5.9.2 (Clang 7.0 (Apple), 64 bit) for MacOS;

- Modern OpenGL Shading Language (GLSL), which is implemented in Qt wrapper and native function; and
- Modern C/C++ programming language.

5. EXPERIMENTS

5.1 Parameters

The parameters that used are a similar configuration to related work [2], with some extent parameters, as described:

- The angle θ (for *top-down* direction);
- The angle φ (for *left-right* direction);
- The density *dens* for percentage visited walking area measurement; and
- Parameters for Genetic Algorithm, as described:
    - The seed for initiating RNG to generate the initial chromosomes;
    - The chromosome length;
    - The total of uniqueness;
    - The total of the population; and
    - The fitness functions.

5.2 Environments

The environments that are used are a similar condition to [2]. Additionally, for 3D exploration for flying agents, we scale *z*-axis to 3:1 for paths. That will affect the calculation for the shortest path distance.

5.3 Performance Measurement

In this paper, we use three factors to measure our approach:

1. The coverage area is calculated by dividing the area visited by the total area that can be visited. Here, we use density *dens* as a calculation area, so if the point visited is in the range, we count as one point. In the case of [1], we did not calculate the coverage area;
2. The execution time starts counting when the pathfinding algorithm is executed until the end of algorithm works. In the case of [2] and GA, we counted until the exploration task was completed, but for [1] we only counted until we found the destination; and,
3. The shortest path distance is calculated using *Euclidean distance*, where for the walking agents use *x* and *y*-axes only. Then, for the flying agents, we use the *z*-axis (scaled), too.

5.4 Testing Scenario

The testing scenario is slightly different from [2]. First, we do for *left-right* growth GA and the stochastic GA. Second, we include the *top-down* growth rules for GA with first termination rules only. We run the test five times, for each scenario. Then, we compare the first scenario with the results [2].

The fitness score (termination) rules used:

1. Greater than 0.8; and,
2. Greater than 0.8 with selected shortest path score of at least 85% from the global shortest path.

The comparison parameters used are described in subsection 5.3 Performance Measurement. To get similar results from [2], we run the same experiments. Also, we compare with stochastic Extended L-Systems [1]. All configuration methods are run five times.

5.5 Results

Table 1. Results for *left-right* growth rule

Configurations	Exec. Time (s)		Shortest path		Coverage Area (%)		Num. of Iterations	
	Mean	Std. dev	Mean	Std. dev	Mean	Std. dev	Mean	Std. dev
Non-flip 45	0.11	0.02	234.02	48.15	**98.03%**	**0.24%**	n/a	n/a
Non-flip -45	**0.08**	**0.01**	**212.49**	**0**	97.14%	0%	n/a	n/a
Flip -45	***0.06***	***0***	*n/a*	*n/a*	***45.15%***	***0%***	*n/a*	*n/a*
Flip Keep -45	0.1	0.01	**180.15**	**23.27**	98.44%	0.07%	n/a	n/a
Keep Flip -45	0.1	0.01	190.56	46.55	97.81%	0.17%	n/a	n/a
Ext-LS (pa=0.33; pb=0.33; pc=0.33)	2.25	0.12	203.2	5.36	n/a	n/a	n/a	n/a
Ext-LS (pa=0.81; pb=0.09; pc=0.09)	**0.45**	**0.06**	199	5.34	n/a	n/a	n/a	n/a
Ext-LS (pa=0.09; pb=0.81; pc=0.09)	1.72	0.08	246	2.92	n/a	n/a	n/a	n/a
Ext-LS (pa=0.09; pb=0.09; pc=0.81)	3.2	0.15	**190.2**	**0.45**	n/a	n/a	n/a	n/a
GA 0.8	7.24	1.5	**179.35**	**8.91**	**97.14%**	**0.62%**	18.8	3.83
GA 0.8 & 85%	**6.81**	**2.4**	180.88	12.01	85.69%	27.02%	**17.4**	**5.94**
PD GA 0.8	**7.98**	**2.43**	165.08	7.84	97.42%	0.8%	**20**	**5.43**
PD GA 0.8 & 85%	8.12	2.22	165.25	9.7	**98.06%**	**0.67%**	20.6	5.68
PDP GA 0.8	9.05	2.61	171.6	19.05	96.72%	2.59%	22.6	6.88
PDP GA 0.8 & 85%	9.49	2.75	**164.14**	**4.75**	97.74%	0.82%	24.4	7.02

The parameters that are used in these experiments are similar to [2], with the addition of:

- $\theta = 45°$ (for "*left-right* and *top-down* growing" only);
- $\varphi = 45°$;
- $dens = 10$ (for "*left-right* growing" only);
- GA's seed $= 512$*;*
- GA's chromosome length $= 6$ (for *left-right* growing only) and $= 8$ (for *left-right* and *top-down* growing only);
- GA's uniqueness $= 10$;
- GA's total population $= 100$; and,
- GA's fitness rules as described in the testing scenario.

For Table 1 in the Configuration column, the elements are described as:

- The "Non-flip," "Flip," "Flip Keep," and "Keep Flip" are a term from [2], where entails with φ = -45° or 45°;
- The "GA 0.8" is the *left-right* growth with the first termination rule;
- The "GA 0.8 & 85%" is the *left-right* growth with the second termination rule;
- "PD" stands for probabilistic degree of branching;
- "PDP" stands for probabilistic degree and branching position;
- The "Ext-LS" is stochastic Extended L-System [1], while pa, pb, and pc are probability value refer to [1] and pa + pb + pc ≈ 1;
- For "execution time," "shortest path" and "number of iterations," lower is better; and,
- For "coverage area," higher is better.

5.5.1 Scenario 1: left-right Growth

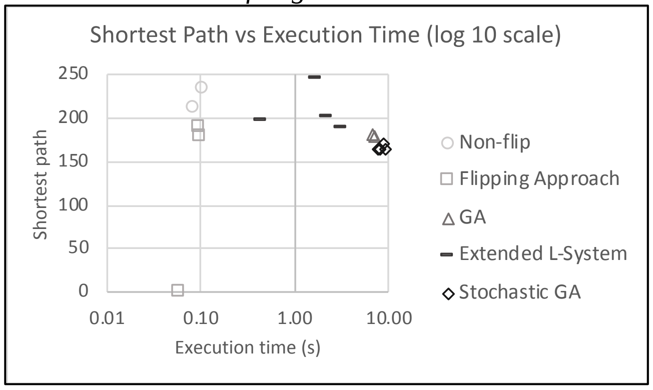

Figure 4. Shortest Path vs Execution Time (s)[1] Comparison Chart.

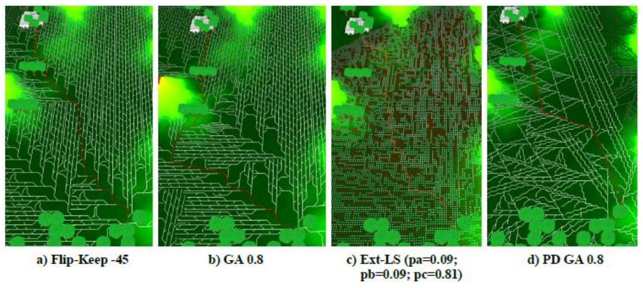

Figure 5. Visualization of L-System's Explorations.

First, considering the results for GA. The results can be seen in Table 1 for "GA 0.8" and "GA 0.8 & 85%." In the manner of time, the mean and standard deviation of the second termination rule is better than the first termination rule – however, in the way of the distance the second termination rule wins.

Then, comparing these results with the *flipping approach* [2] shows that GA is slower, but GA produces a better shortest path solution. Also, when comparing with stochastic Extended L-System [1]. We can see that *flipping approach* and GA are slightly better. Respectively the shortest path results are "Flip Keep –45" = 180.15, "GA 0.8" = 179.35, and "Ext-LS (pa=0.09; pb=0.09; pc=0.81)" = 190.2.

Besides, we compare with stochastic GA results. Stochastic GA is the slowest, but the shortest path is significantly affected. The trade-off for this method is "PD GA 0.8," with a result of 165.08 using the execution time of 7.98s. This result is 15 points closer and only 1 second slower than "GA 0.8 & 85%". Then when we compare with the shortest results of [1], "Ext-LS (pa=0.09; pb=0.09; pc=0.81)". "PD GA 0.8" is 25 points shorter and only 2.3 times slower.

The trade-off results between the shortest path distance and the execution time can be seen in Fig. 4, where we classified as "Non-flip," "Flipping Approach," "GA," stochastic "Extended L-System" and "Stochastic GA" ("PD" and "PDP").

Fig. 5 shows the visualization results of "Flip Keep -45," "GA 0.8," "Ext-LS (pa=0.09; pb=0.09; pc=0.81)," and "PD GA 0.8." The red line is the solution path and the white line, slowly turning dark, is the exploration path. The target represented by the white building and the starting point is at the other end of the red line. For Fig. 5c, the path is described as white and red dots, where red dots mean removed path (death rules).

5.5.2 Scenario 2: left-right and top-down Growth

In this scenario, we consider 3D GA for flying agents. The results can be seen in Table 2, where each chromosome contains eight genes. Eight is selected because the degree of freedom of direction is 4, and there are two actions, *flip* and *keep*. We can see, that execution time is directly proportional growth with the number of iterations.

6. CONCLUSIONS

L-System and Extended L-System for pathfinding with GA can provide a reasonable solution for exploration and path creating. In this study, we show that our method can be used to make a combination pattern of *flip* and *keep* actions [2] without defining all possible combinations. Not only can it be used for 2D exploration but also 3D exploration for flying agents. Regarding distance, GA can contribute — however the GA makes execution time longer.

Overall, our approach is successful for exploration and creating multiple-path simultaneously. However, in some cases, not all possible areas are visited.

In this paper, only one heightmap[2] has been used. In the future work, we plan to use several heightmaps and classified a pattern based on the surface layout. Further, we plan to develop event-based *flip* and *keep* to reduce execution time.

[1] Scaled in Log 10.

[2] http://www.rastertek.com/pic4005.gif

7. ACKNOWLEDGMENTS

This research was supported by the Japan International Cooperation Agency – JICA under Innovative Asia program, Grant No. D1707473. We thank our colleagues from JICA Chubu International Center who provided insight and expertise that greatly assisted the research, although they may not agree with all of the interpretations/conclusions in this paper.

8. REFERENCES

[1] Ishida, Y. 2017. Path-finding and path-finding system based on the extended L-system. *Innovation Japan*.

[2] Djusdek, D. F. and Ishida, Y. 2018. 3D L-Systems Path-finding Case study: Walking Agent's Explorations in Simulations. *The 32nd Annual Conference of the Japanese Society for Artificial Intelligence*.

[3] Man, K. F. 1996. Genetic Algorithms: Concepts and Applications. *IEEE Transactions on Industrial Electronics* 43, 5 (Oct. 1996), 519-534.

[4] Ochoa, G. 1998. On genetic algorithms and lindenmayer systems. *International Conference on Parallel Problem Solving from Nature V*. Springer, Amsterdam, 335-344.

[5] Bernard, J. and McQuillan, I. 2017. New Techniques for Inferring L-Systems Using Genetic Algorithm. arXiv:1712.00180v2. Retrieved from https://arxiv.org/abs/1712.00180.

[6] Dray, T. and Manogue, C. A. 2002. Conventions for Spherical Coordinates. Oregon State University.

[7] Matsumoto, M. and Nishimura, T. 1998. Mersenne twister: a 623-dimensionally equidistributed uniform pseudorandom number generator. *ACM Trans. on Modeling and Computer Simulation* 8, 1 (Jan. 1998), 3-30.

Table 2. Results for *left-right* plus *top-down* Growth Rule with Termination Rule 1

	Test					Mean	Standard deviation
	1	2	3	4	5		
Chromosome	+&-^+&^&	&&-&^^-+	^&&&+-+^	&^+-&---	&^^++&^&	-	-
Iteration	15	28	26	36	18	24.6	8.35
Execution time (s)	12.91	28.07	28.07	44.26	17.55	26.17	12.09
Shortest Path	189.6	206.38	200.89	190.73	198.31	197.18	7.05