ISSN 2085-4552

Creating a Randomized Residential Cluster Using Neural Network Method

Tirta Prawisuda, Albert Van Otto, Taufiq Radityadji

Computer Science Department, Universitas Multimedia Nusantara, Tangerang, Indonesia

Accepted on December 9, 2013
Approved on December 23, 2013

***Abstract*—Along with the city growth around the world, residential cluster also grows as a more preferable choice of housing. The high amount of residential cluster makes it inevitable for each cluster to looks the same with the other cluster. This paper will try to develop an idea so each cluster will look different.**

***Index Terms*—Artifical intelligence, computing, neural network, residential cluster.**

I. INTRODUCTION

People of this era apparently realize that there is less land available for rural development. This realization leads them to come up with the idea of putting together a group of houses into the same area while still having open spaces around them. Even so, the widespread of this method cause the inevitable effect of each cluster looks the same with another cluster. With that problem in mind, we decide to come up with a concept where an AI will create a randomized residential cluster with neural network method.

II. RELATED WORKS

Our work is heavily influenced by 2D maze procedural generator [1]. The idea was to generate a map based on some parameters and generate the area based on the array procedurally with the method defined.

III. METHOD

This section will focus about the steps involved in creating the residential cluster and the concept for the neural network [2] that consists of three parts: input neuron, hidden neuron, and output neuron. Our approach this time will only consider the first development without considering what will happen next (e.g., rebuilding because of natural disaster).

The generation process is divided into three parts where in the first part; user will be prompted to insert their desired number for the land area and convert it into blocks with the size of both width and height is ten meter. Then, the AI will utilize its hidden neurons to generate the residential cluster and give the finished map as an output.

The artificial neural network the AI has is divided into three layers just like stated above. The first layer which is the input layer is necessary so that they can receive input from outer region. The second layer is the hidden layer that sits between the input and the output which has no interaction with the environment. The final layer is the output layer which interacts with the output to present the processed data. Our work here will have two input neurons, four hidden neurons, and one output neuron.

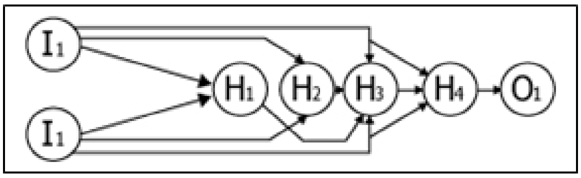

Fig. 1. The design for the AI's ANN

There are a few things that we will try to create through function inside he hidden layer and those are:

- Generating the street,
- Amount of houses,
- Location of said houses,
- Generating parks.

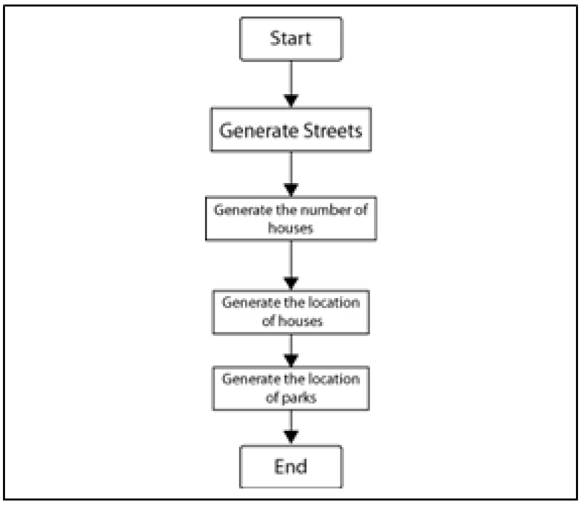

Fig. 2. Flowchart of the process

A. Generating Streets

Before the process starts, the system will ask for two inputs from the user which are width and length of the land. After the system receives information about the area size, it will turn the total area into some 10x10 meter blocks. The system then proceeds to spread an imaginary map using those blocks and turns them all into street blocks.

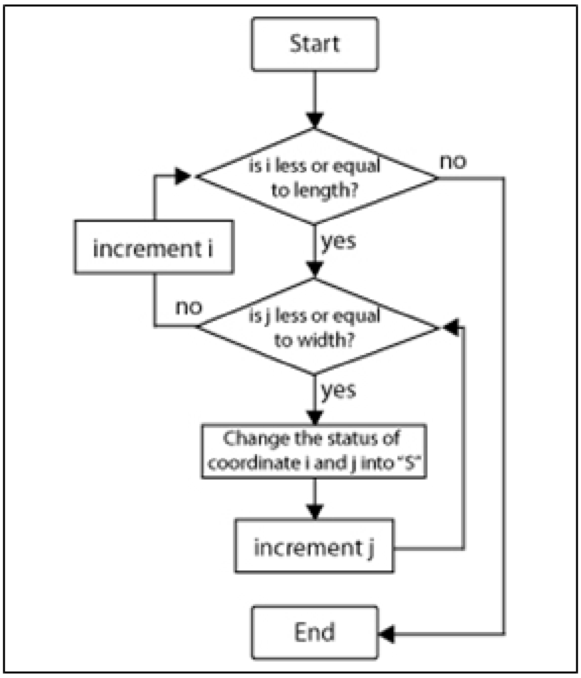

Fig. 3. Flowchart of the first process.

B. Generating the Number of Houses

The second step is to create a number of 10x10 meter house blocks while taking the size of the area into consideration. The amount of houses will be divided into two which are minimum and maximum. The minimum amount of houses is 25% of total area while the maximum amount of houses is 50% of total area.

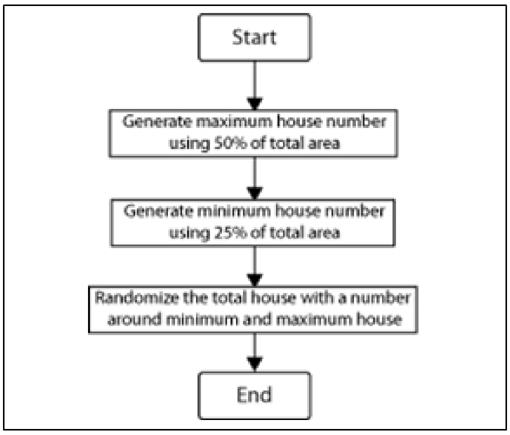

Fig. 4. Flowchart of the second process.

C. Generating the Location of Houses

With the total amount of houses now known, the third process will decide where to place all the house blocks. Before the system do anything, it will first check to a temporary variable whether it exceeds the number of the total house. If it does not exceed the number, it will proceed to randomize a number between one and the number of total area size. If the area is already filled with a house block, it will call itself. Else if the block is still a street block, it will turn the street block into a house block and call itself after incrementing the temporary variable used to see how much house done being generated. This process will continue to loop until the amount of house is fulfilled.

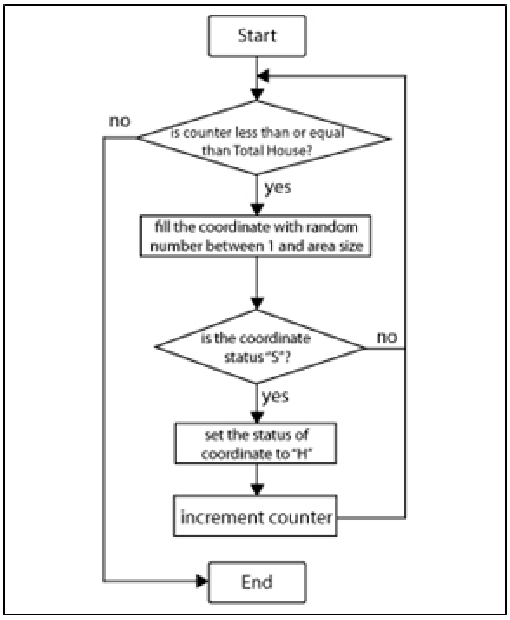

Fig. 5. Flowchart of the third process.

D. Generating Parks

The final step will be to create a park for the open space. This step will be done by the final process that will first randomize the amount of the park with the maximum number being 40% of the total area size. Before doing anything, the system will check a temporary variable whether it exceeds the number of total park to be generated. If it does not exceed the number of the total park it will proceed to create a random number to be used for coordinates. If the status of the coordinate is either a park block or house block, the system will loop itself to before the part where it creates a random number. If the status of the coordinate is simply a street block, the system will turn it into a park block. After turning the block into a park block, the system will increment a temporary variable used to store the amount of park blocks that has been generated. The system will then loop itself back to the process where it checks the temporary variable and decide whether to continue this process or terminate it because the desired number of park is already met.

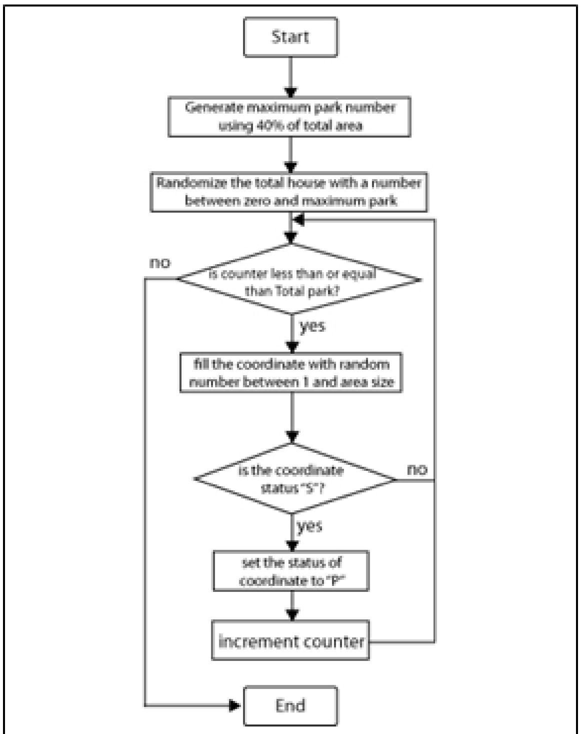

Fig. 6. Flowchart of the final process.

IV. TESTING

For the testing of our system, we will include 4 scenarios which are:

1. Houses that don't have street around them,
2. Houses with no park around,
3. Park with no road, and
4. House with both park and road.

Our testing was performed 400 times and the area we used was 10 meter in both length and width. The following are the result of the first 50 samples of the test we conducted.

TABLE I. RESULT OF THE FIRST 50 TESTS

No	H	S	P	Scene 1	Scene 2	Scene 3	Scene 4
1	40	27	33	16	6	10	18
2	27	51	22	3	11	2	14
3	25	63	12	1	16	0	8
4	45	40	15	2	22	4	21
5	44	53	3	2	40	0	4
6	33	65	2	1	32	0	1
7	45	18	37	26	7	14	13
8	42	45	13	5	20	4	18
9	39	59	2	1	36	0	3
10	42	39	19	7	20	3	21
11	39	50	11	3	22	1	14
12	37	55	8	4	30	0	6
13	47	18	35	22	10	17	19
14	36	40	24	6	14	3	18
15	40	30	30	9	8	6	24
16	50	26	24	16	17	10	22
17	33	57	10	1	19	4	13
18	37	43	10	2	19	4	16
19	28	41	31	5	5	13	18
20	40	57	3	1	34	0	5
21	46	35	19	10	22	4	17
22	29	32	39	10	4	9	17
23	29	38	33	6	8	7	15
24	29	64	7	2	26	0	1
25	34	43	23	8	13	5	16
26	45	21	34	21	13	14	15
27	35	30	35	8	8	13	19
28	31	52	17	6	20	1	9
29	37	56	7	3	27	0	8
30	29	32	39	5	4	11	20
31	49	47	4	4	42	1	7
31	25	62	13	0	15	1	10
33	33	65	2	0	31	0	2
34	40	40	20	2	20	1	18
35	26	38	36	5	4	6	17
36	30	58	12	0	16	0	14
37	36	44	20	3	23	3	13
38	47	24	29	17	15	13	21
39	32	51	17	7	16	1	10
40	28	61	11	0	15	0	13
41	47	34	19	8	24	3	19
42	44	54	2	1	41	0	2
43	33	56	11	2	22	0	9
44	30	40	30	7	5	4	18
45	45	48	7	6	36	0	7
46	31	31	38	7	3	9	21
47	27	37	36	4	5	4	18
48	39	58	3	0	36	0	3
49	32	50	18	3	15	4	14
50	38	61	1	2	37	0	1

After we did the test, we can achieve these average percentages:

TABLE II. AVERAGE PERCENTAGE OF EACH SCENARIO

Scenario 1	17.17196%
Scenario 2	47.86715%
Scenario 3	21.33671%
Scenario 4	37.30400%

These percentages shows that the probability of the houses being built and have no streets around them is 17.17196% and it goes the same with the second to fourth scenario. We realize that these results won't compare to the condition on the field, but this should give a clear image on the accuracy of the system.

V. RESULT AND CONCLUSION

400 testing gave us different result each time. Even so, we can also see that there are times where the numbers are almost close if not the same. Despite the same number of items, the location will always be unique due to the randomization on the coordinates. This proves that the system is capable to create a different residential cluster each time.

The accuracy depicted on the table from session IV is based solely on the fulfillment of each scenario goal. For example in scenario one, we were searching for houses that have no street around them. On the testing, we focus on the house that has no street around, oblivious to the fact that the house could have a street yet there is no way for the inhabitant of the house to get anywhere.

Despite how the result fare and how hard it will be to implement in the actual field for now, our system is proven able to create a randomized residential cluster using a simple artificial neural network philosophy.

We hope that with the advance of the era, there will be solutions to fix the problem that persist on our system as said on the second paragraph.

ACKNOWLEDGEMENT

We would like to thank everyone that has been helping us and supporting us during the time we write this literature. We also would like to thank God for giving us the opportunity to make this happen.

REFERENCES

[1] Daniel Ashlock, Colin Lee, and Cameron McGuinness. "Search-Based Procedural Generation of Maze-Like Levels". September 2011.

[2] Simon Haykin, Neural Networks: A Comprehensive Foundation, 2nd ed., Pearson Education : India, 1999.

[3] Daniel Ashlock, Colin Lee, and Cameron McGuinness. "Search-Based Procedural Generation of Maze-Like Levels". September 2011.

[4] Groenewegen, Smelik, Kraker, and Bidarra, "Procedural city layout generation based on urban land use models", 2009.

[5] Hwang and Thill, "Using fuzzy clustering methods for delineating urban housing submarkets", November 2007.

[6] Lokasyuk, Pomorova, and Govorushchenko, "Neural nets method for estimation of the software retesting necessity", May 2008.