Combining Procedural Generation and Genetic Algorithms to Model Urban Growth

Etienne Tack[1,2][a], Gilles Énée[2][b] and Frédéric Flouvat[3][c]
[1] *INSIGHT SAS, Noumea, New Caledonia*
[2] *University of New Caledonia, ISEA, Noumea, New Caledonia*
[3] *Aix-Marseille Univ., CNRS, LIS, Marseille, France*
fl

Keywords: Procedural Generation, Genetic Algorithm, Agent-Based Modelling, Urban Growth.

Abstract: In this paper, we present an approach to model spatial influences in multi-agent models using procedural generation and genetic algorithms. We applied this approach in an urban growth model. In agent-based simulations, the agents make decisions based on the perception of their environment. In our context, the agents represent inhabitants who can create new buildings or extend the existing ones. Their behaviour is ruled by spatial influences (e.g., the proximity of the road increases chances of building in the surrounding areas). Procedural generation provides a good framework for representing the influences of the environment on the agent's behaviour. Each spatial feature is associated with an influence function. Their parameters search space can be tremendous, making it difficult for field experts to set them manually. Consequently, we use a genetic algorithm to optimize the parameters of these influence functions and train the model based on three spatial measures (Chamfer distance, kernel density, and a density grid). This approach can be employed likewise to any problem where the agent decisions are wholly or partly based on location. Our experiments highlight the interest of our approach and the impact of the chosen fitness functions.

1 INTRODUCTION

Urban growth poses significant challenges, particularly in developing regions where informal settlements often emerge without strict planning rules. These areas, defined by UN-Habitat as lacking secure tenure, basic infrastructure, and adherence to building regulations, are shaped by unspoken traditions and geographical constraints. Existing urban growth models, including multi-agent systems, often assume rigid spatial structures like grid layouts, making them poorly suited for informal contexts.

Multi-agent systems have been widely used to simulate urban dynamics by modelling individuals or households as agents. While grid-based approaches (Schelling, 1971; Barros, 2004; Zhang et al., 2010; Jokar Arsanjani et al., 2013; Schwarz et al., 2016; Picascia and Yorke-Smith, 2017; Agyemang et al., 2022) simplify implementation, they rely on strong assumptions, such as fixed spatial resolution and uniform building characteristics, limiting realism. More precise vector-based approaches (Augustijn-Beckers et al., 2011) better reflect spatial variation but often embed restrictive local assumptions about construction patterns. In computer graphics, procedural generation has also been applied to create virtual cities (Smelik et al., 2014), yet these approaches rarely validate spatial accuracy beyond visual assessments.

Addressing these gaps, this study proposes a novel method combining procedural generation and genetic algorithms to model spatial dynamics in agent-based urban growth simulations. Inspired by Tobler's first law of geography (Tobler, 1970), which emphasizes proximity effects, we define two types of spatial influence functions and optimize their parameters using NSGA-II. Our approach eliminates the need for empirical assumptions and improves realism by learning from spatial data. Experiments demonstrate the applicability of this method in generating realistic urban growth patterns, validated through measures like density differences and Chamfer distance.

[a] https://orcid.org/0000-0003-4131-1449
[b] https://orcid.org/0000-0002-0140-5291
[c] https://orcid.org/0000-0001-7288-0498

Tack, E., Énée, G. and Flouvat, F.
Combining Procedural Generation and Genetic Algorithms to Model Urban Growth.
DOI: 10.5220/0013165200003890
Paper published under CC license (CC BY-NC-ND 4.0)
In *Proceedings of the 17th International Conference on Agents and Artificial Intelligence (ICAART 2025) - Volume 1*, pages 420-428
ISBN: 978-989-758-737-5; ISSN: 2184-433X
Proceedings Copyright © 2025 by SCITEPRESS – Science and Technology Publications, Lda.

2 THE INFLUENCE SUB-MODEL

Usually, agents are influenced by their local perception of the world. In our work, we use influence functions to determine where agents may construct new buildings at each iteration of the simulation. An influence function is defined, with the help of an expert (geographer), for each environmental factor that is related to construction of new buildings.

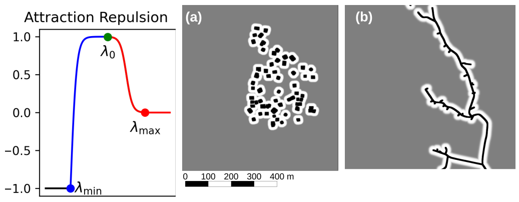

Figure 1: Attraction-Repulsion influence function and its application to generate influence maps for buildings (a), roads (b) and paths (c). (White = 1, Grey = 0 and Black = −1).

For example, the attraction-repulsion function in Figure 1 (first plot) illustrates the influence of existing buildings or paths on the construction of a new building. In this figure, x represents the distance between a candidate position for a new building and the nearest existing building or path, and y its influence on its construction. Figure 1(a) shows the influence map of all existing buildings based on this function. The black areas (positions of existing buildings) are repulsive, and the white areas are attractive. This is because new buildings cannot be built on top of each other, but they tend to be close together (like a village). Figure 1(b) represents the influence maps of paths extrapolated from this same influence function.

Based on Figure 1, the attraction repulsion function can be more formally and more generally defined as:

$$AR(x) = \begin{cases} -1 \text{ if } x < \lambda_{min} \\ f(x) \text{ if } \lambda_{min} \leq x < \lambda_0 \\ g(x) \text{ if } \lambda_0 \leq x < \lambda_{max} \\ 0 \text{ if } x \geq \lambda_{max} \end{cases} \quad (1)$$

where λ_{min}, λ_0 and λ_{max} are three distance thresholds ($\lambda_{min} < \lambda_0 < \lambda_{max}$).

In equation 1, $f(x)$ represents the increase in influence from repulsive to attractive and $g(x)$ represents the decrease in influence from attractive to neutral. To define these two functions, we use the hyperbolic tangent (tanh). It has been slightly modified to control the slope of the curve and the length of the transition between high repulsion (i.e. -1) and high attraction (i.e. 1), as a basis for $f(x)$ and $g(x)$.

More precisely, these functions are defined by:

$$\begin{aligned} f(x) &= \tanh\left((x-\lambda_{min}) \times \frac{2\pi}{\lambda_0 - \lambda_{min}} \right) * 2 - 1 \\ g(x) &= -\tanh\left((x - \frac{\lambda_{max}+\lambda_0}{2}) \times \frac{2\pi}{\lambda_{max}-\lambda_0} \right) \\ & * \frac{1}{2} + \frac{1}{2} \end{aligned} \quad (2)$$

Of course, different types of factors have different influences, i.e. potentially multiple influence functions can be identified by expert and used in our approach. For instance, Figure 2 illustrates the influences of paths and slope as defined by our expert. Such influence is based on the open distance function illustrated in Figure 2.

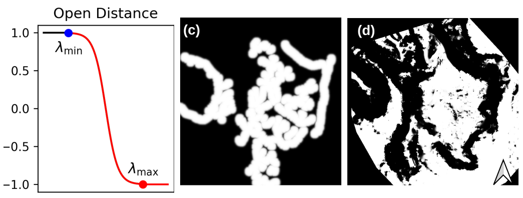

Figure 2: Open Distance influence function and its application to generate the influence maps of the paths (c) and the slope (d). (White = 1, Grey = 0 and Black = −1).

The open distance function can also be defined using the same approach. Its formula is given below:

$$OD(x) = \begin{cases} 1 \text{ if } x < \lambda_{min} \\ h(x) \text{ if } \lambda_{min} \leq x < \lambda_{max} \\ -1 \text{ if } x \geq \lambda_{max} \end{cases}$$

$$h(x) = -\tanh\left((x - \frac{\lambda_{max}+\lambda_{min}}{2}) \times \frac{2\pi}{\lambda_{max}-\lambda_{min}} \right) \quad (3)$$

Once all influence maps are processed, they are weighted and aggregated into a single influence map using the same equation introduced by (Emilien et al., 2012).

$$I(P) = \begin{Bmatrix} -1 \text{ if } \exists k \text{ such that } I_k(P) = -1 \\ \max(0, \sum w_k.I_k(P)) \text{ in other cases.} \end{Bmatrix}. \quad (4)$$

where P is a candidate location, $I_k(P)$ is the attraction value of the location P according to factor k in the corresponding influence map, and w_k the weight for each influence factor k.

Figure 3 shows the results of the aggregation of the four influence maps shown in Figures 1 and 2 (rendered at a resolution of 1 m). The parameters λ_{min}, λ_0 and λ_{max} of these influence functions, as well as the weights w_k, are defined by reinforcement learning as explained in the next section.

Finally, a gradient descent is done, based on this aggregated influence map to find the most attractive

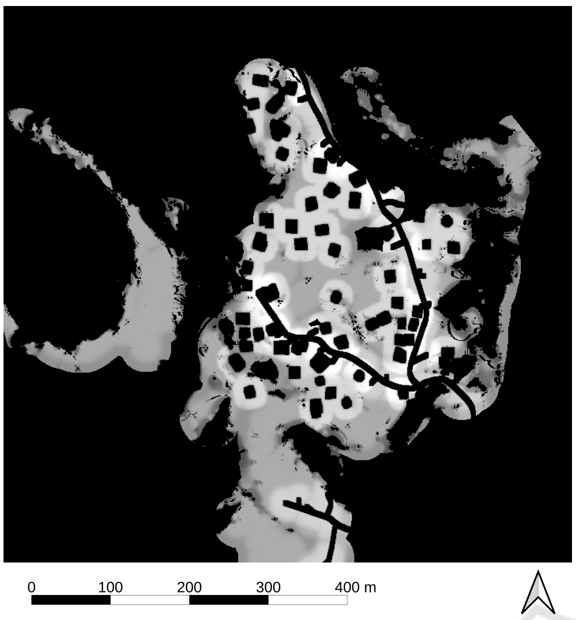

Figure 3: Aggregated Influence Map.

spots on the map without having to calculate the whole map, which would be computationally expensive as the map needs to be updated after every modification (e.g., adding a building).

3 DETERMINING PARAMETERS OF INFLUENCE FUNCTIONS USING A GENETIC ALGORITHM

As previously explained, the influence sub-model is defined by a set of influences functions, their parameters (λ_{min}, λ_0 and λ_{max} in Equations 1-3) and their weights for the aggregation function (i.e. the w_k parameters in equation 4). To optimize these parameters, the genetic algorithm NSGA-II (Deb et al., 2002) has been chosen. It stands for "Non-dominated Sorting Genetic Algorithm II" and it is widely used as a multi-objective optimisation algorithm.

3.1 Validation Measures

Genetic algorithms need fitness functions to evaluate the accuracy of candidate solutions (in our case, parameters of the influence functions). As displayed in Figure 4 (Crooks, 2018), measures compare the data generated by the multi-agent model against observed data. Then, the model is tuned, run and evaluated until error is sufficiently low. Because the outcomes of the simulations are a spatial representation of our model, measures that take account of this spatial dimension are needed. Three spatial measures have been chosen: a grid-based density difference to verify that the overall placement of the building is more or less the same, the Chamfer distance to quantify and summarise errors into a single value, and a kernel density difference to show where the difference of density of buildings are.

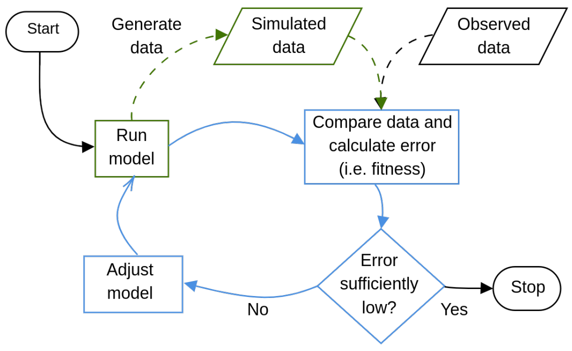

Figure 4: The process of validation and calibration.

The grid-based density difference measure estimates building density differences between simulated and real-world data, based on a grid decomposition of space. As illustrated in Figure 5, space is sliced in a fixed number of cells. Each cell represents a different area, with a certain number of buildings within it. The centroid of the building is used to determine the cell it belongs to. Thus, we obtain a density matrix representing the number of buildings in each area. The grid-based density difference is the absolute difference between the grid-based density matrix of real-world data and the one of simulated data.

Formally, given two grid-based density matrices A and B, each value of the grid-based density difference d_D is given by:

$$d_D(A,B) = \sum_{i,j} |A_{ij} - B_{ij}|$$

where A_{ij} and B_{ij} are the number of buildings within the cell (i, j).

The Chamfer distance is another validation measure considered in this work. It measures the difference between two point clouds. It sums the squared distances for each point with its nearest neighbour in the reference data set (see Figure 6). More formally, for two non-empty subsets X and Y, the Chamfer distance $d_C(X,Y)$ is:

$$d_C(X,Y) = \sum_{x\in X} \min_{y\in Y} ||x-y||^2 + \sum_{y\in Y} \min_{x\in X} ||x-y||^2$$

This measure will be applied to the centroids of the buildings, and will give more information about local differences.

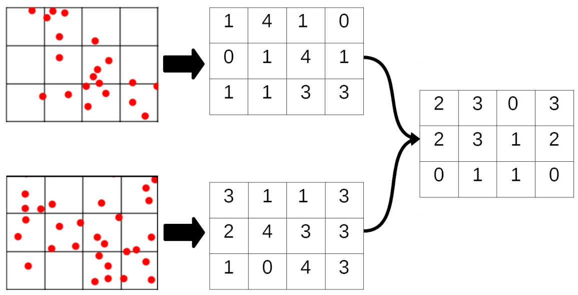

Figure 5: Example of density grid processing. The two first figures represent two spatial distributions of buildings (each red point represents the centroid of a building). The two matrices in the middle represent the number of buildings in the corresponding cell. The last matrix is the absolute difference of the two previous density matrices.

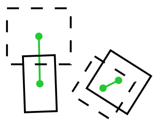

Figure 6: Chamfer distance example (dotted shapes are from the validation dataset, distances are calculated between centroids).

The last validation measure studied is the kernel density difference. It is simply the difference of two kernel density estimations. In a GIS context, kernel density should be more expressive/relevant than point density, since it will smooth the density at one point by looking at the influence of points all around. More formally, the kernel density estimation at given coordinates (x,y) is given by

$$\hat{f}(x,y) = \frac{1}{n \cdot h^2} \sum_{i=1}^{n} K\left(\frac{x-x_i}{h}\right) \cdot K\left(\frac{y-y_i}{h}\right)$$

where:

- n is the number of points,
- h is the bandwidth, which smooth the density estimate,
- $K(u)$ is the kernel function, which is a symmetric and non-negative function that integrates to 1 (in our work, we used the Cosine kernel),
- x_i and y_i represent the coordinates of the ith point in the input data set.

We process kernel density estimations for simulated data and real-world ones. Then, we calculate the difference between the two generated heat maps. Figure 7 shows the kernel density difference between two maps. Red areas represent a lack of buildings in the simulation, and blue areas shows the contrary. If the density is identical, it is displayed in white. To use this measure as a fitness function for the genetic algorithm, we take the average of absolute differences for each pixel in both heatmaps, i.e.:

$$d_K(A,B) = \frac{1}{wh} \sum_{i=1}^{w} \sum_{j=1}^{h} \left| \hat{f}_A(i,j) - \hat{f}_B(i,j) \right|$$

where:

- $\hat{f}_A$ is the kernel density over the buildings centroids generated by the simulation,
- $\hat{f}_B$ is the same but for the control dataset,
- (w,h) is the size of the bounding box of the two combined datasets.

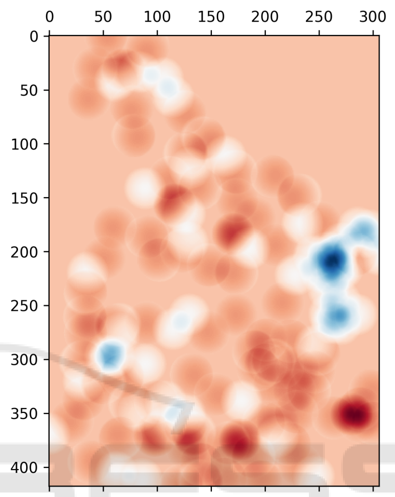

Figure 7: Example of Kernel Density Difference obtained for an informal settlement located in Fiji.

4 EXPERIMENTS

4.1 Case Study

Our first case study is an informal settlement located near Suva, Fiji (its size is $118,885\ m^2$). As summarised in Table 1, three spatial features (buildings, roads and paths) of this settlement have been extracted by experts using visual interpretation for four dates (1994, 2002, 2009 and 2019).

Table 1: Spatial Features and Data Sources

Date	**Image Type**	**Image Res.**	**Extracted Features**
1994	Aerial	$2.3m$	61 buildings
2002	Satellite	$0.8m$	83 buildings and 31 roads/paths
2009	Satellite	$0.59m$	103 buildings and 46 roads/paths
2019	Aerial	$0.35m$	129 buildings and 158 roads/paths

To integrate the slope into the model, a digital elevation model (DEM) has been interpolated from a digital surface model (DSM) produced during a drone image acquisition in 2019.

4.2 Experiment Setup

Our code is developed in Python, and is available on a GitHub repository[1], along with all the data used for the experiments. The informal settlements agent-based model is built on top of MESA (an agent-based modelling framework), and we used PyMOO (Blank and Deb, 2020) for optimizing the influence function parameters. It is a Python package that implements multi-objectives optimisation algorithms.

As shown in the Table 2, our expert identified two types of influence functions:

- an attraction-repulsion function for buildings and roads;
- an open distance function for paths (because buildings can be built on paths);
- an open distance function for slope (because a flat area is very attractive, while sloping ones are not). Note that this function do not take a distance as input, instead it takes the slope under the submitted building.

Table 2: Genetic algorithm's decision variables and constraints, associated with the influence functions.

Influence	Parameters	Constraints
Buildings	$\lambda_{\min} = X_0$	$0 \le X_0 \le 100$
(Attraction-	$\lambda_0 = X_0 + X_1$	$0 \le X_1 \le 100$
Repulsion)	$\lambda_{\max} = X_0 + X_1 + X_2$	$0 \le X_2 \le 100$
	$w = X_3/(X_3 + X_7 + X_{10} + X_{13})$	$0 \le X_3 \le 1$
Roads	$\lambda_{\min} = X_4$	$0 \le X_4 \le 100$
(Attraction-	$\lambda_0 = X_4 + X_5$	$0 \le X_5 \le 100$
Repulsion)	$\lambda_{\max} = X_4 + X_5 + X_6$	$0 \le X_6 \le 100$
	$w = X_7/(X_3 + X_7 + X_{10} + X_{13})$	$0 \le X_7 \le 1$
Paths	$\lambda_{\min} = X_8$	$0 \le X_8 \le 100$
(Open	$\lambda_{\max} = X_8 + X_9$	$0 \le X_9 \le 100$
Distance)	$w = X_{10}/(X_3 + X_7 + X_{10} + X_{13})$	$0 \le X_{10} \le 1$
Slope	$\lambda_{\min} = X_{11}$	$0 \le X_{11} \le \pi/2$
(Open	$\lambda_{\max} = X_{12}$	$0 \le X_{12} \le \pi/2$
Distance)	$w = X_{13}/(X_3 + X_7 + X_{10} + X_{13})$	$0 \le X_{13} \le 1$
		$X_{11} \le X_{12}$

Default settings of PyMOO's NSGA-II implementation has been used. It involves a polynomial mutation (set with a perturbation index to 20) and a simulated binary crossover (with distribution index set to 15 and the probability that it will happen set to 0.9). NSGA-II population consists of 50 individuals, and the genetic algorithm stops when one of the following termination criteria is met : the number of 300 generations is reached, or the objectives values are stable for the last 30 generations.

To find the most appropriate fitness function for NSGA-II, seven combinations of validation measures have been tested on the three time periods of our case study (1994 to 2002, 2002 to 2009, 2009 to 2019). The grid-based density difference is computed using a spatial resolution of $50m$ (each cell represents an area of $50m \times 50m$). NSGA-II was run 50 times for each possible combination of measures and for each period.

[1] github.com/etiennetack/spatial_influences

4.3 Evaluating Learnt Influences

To analyse the results obtained, we first studied the performance of each validation measure combination. To proceed, we compared the resulting influence models against the real data (see Figure 8).

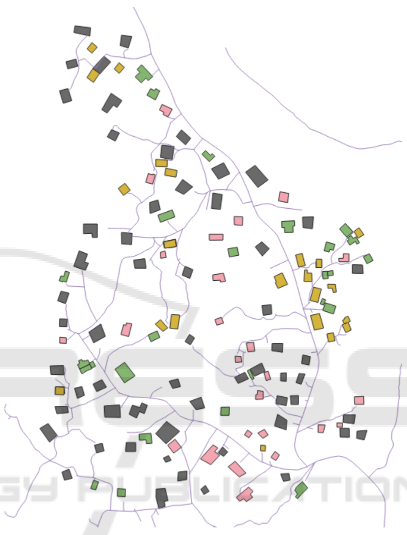

Figure 8: Each colour represents new buildings at each time period: 1994 is grey, 2002 is pink, 2009 is yellow and 2019 is green.

In order to do so, we measured how many new buildings in the field data are placed on areas marked as repulsive by the model (-1). These buildings highlight errors done by the influence sub-model. Therefore, the influence error is obtained by:

$$\text{Influence Error} = \frac{\text{Number of New Buildings On Repulsive Areas}}{\text{Number of New Buildings}}$$

Figure 9 displays the mean influence error for every validation measures combinations (i.e. fitness function in NSGA-II). As a baseline, we also calculated the influence error for random influence function parameters that comply with the same constraints used for the genetic algorithm (see Table 2).

Those results show that NGSA-II was able to optimize the parameters of the influence functions in a spatial agent-based context thanks to the chosen validation measures. On average, the lowest influence

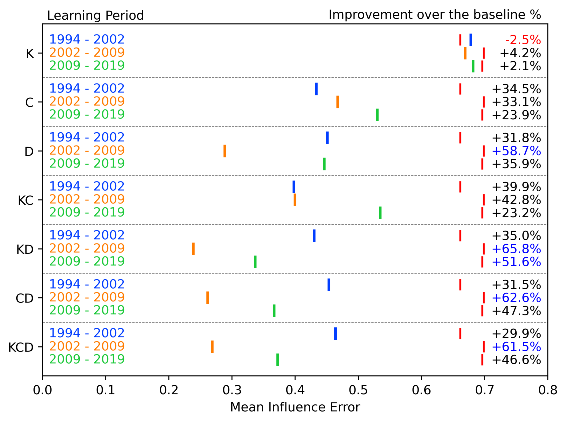

Figure 9: Application of the influence error for each measure combination and for each period.

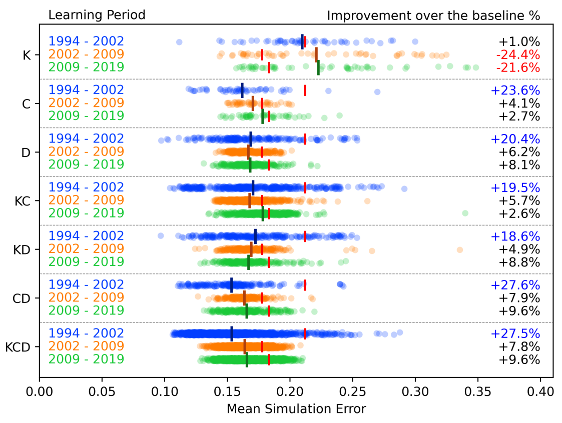

Figure 10: Simulation errors for each measure combination and each period (horizontal bars, represents the error obtained for random simulations and the percentage numbers indicates the difference between the mean simulation error of influence models and random simulations).

error values are achieved using the grid-based density difference measure combined with at least another measure.

4.4 Evaluating the Simulations

Although the influence error gives a good idea about the quality of the influences functions, it does not guarantee the realism of the resulting simulations. Thus, a new error based on the simulation results obtained using each resulting influence function set is needed. A normalised Euclidean distance has been used and is applied between the new buildings added during simulation and the new buildings observed in reality (see Table 1). One specificity when calculating this distance is that when a validation building is selected it is removed from the validation set, in such a way that it cannot be used twice as a reference building.

As a baseline, random simulations had been conducted. During those simulations, buildings are placed randomly within the borders of the area of interest without taking account of any kind of constraint (e.g., buildings can be constructed over other buildings).

As shown in Figure 10, the best improvements over random simulations are still reached using models obtained with the grid-based density difference combined with at least another measure.

4.5 Learnt Models

Figure 11 shows the influence functions obtained with the best combination of measures. Although the solutions are near for the periods of 1994 to 2002 and 2009 to 2019, there is no unique model that works from 1994 to 2019. The influence functions obtained for 2002 are different from the two others. The influence of neighbouring buildings and roads on constructions are low in 2002 (mainly around zero). The influence of paths seems also more important (values are more extreme). In 2002, the slope only has a negative influence when it is close to 2 radians (x-axis), while this value is much lower for 1994 and 2009 (0.5 radians).

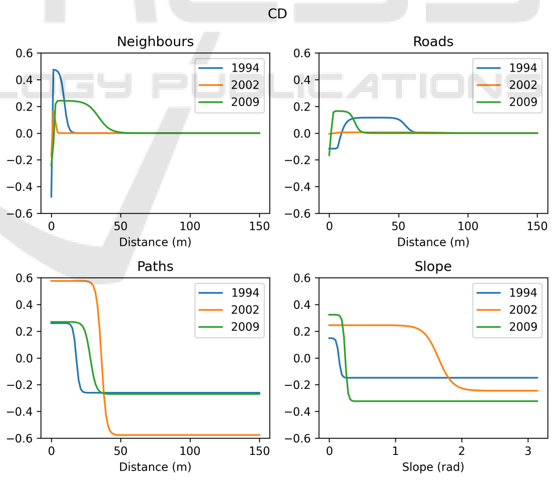

Figure 11: Influence functions for the best solutions obtained with the Chamfer distance and the grid-based density measures.

These results are still very encouraging since this approach attempts to simulate human choices by only considering environmental factors, while there are other factors that influence their behaviour (e.g. socio-economic factors).

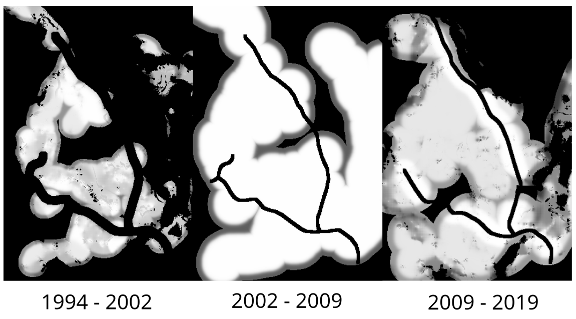

Figure 12: Influence maps of the influence functions displayed in the Figure 11.

5 EXPERIMENTING ON A BROADER DATASET

To evaluate the robustness and generalisability of our proposed urban growth model, we conducted a series of experiments using the SpaceNet 7 dataset (Etten et al., 2021). This dataset offers a comprehensive collection of building footprints derived from high-resolution satellite imagery for various cities all over the world. providing an opportunity to test our approach on a broader and more diverse dataset compared to the controlled datasets used in earlier sections of the study. This gives us larger and more diverse datasets than those presented previously on Suva. For this experiment, we chose data (annotated building footprints) from cities in multiple geographic regions. These regions vary in urban density, development patterns, and environmental features, making this dataset particularly suitable for stress-testing our model's capacity to generalize across different urban morphologies. We also specifically selected areas where the number of building did not decrease over time. The selected datasets, are displayed in Table 3.

Table 3: SpaceNet 7.

SpaceNet 7 Code	Simple Name	Size (sq. km.)	Building Delta
L15-0368E-1245N_1474_3210_13	Los Angeles	15.58	366
L15-0577E-1243N_2309_3217_13	New York	15.83	169
L15-0632E-0892N_2528_4620_13	Santiago	19.58	122
L15-1025E-1366N_4102_2726_13	London	9.37	326
L15-1138E-1216N_4553_3325_13	Tripoli	17.25	1001
L15-1203E-1203N_4815_3378_13	Cairo 1	17.96	1002
L15-1204E-1204N_4819_3372_13	Cairo 2	17.24	240
L15-1296E-1198N_5184_3399_13	Kuwait	18.38	3082

The road network data was extracted from OpenStreetMap (OpenStreetMap contributors, 2017), providing detailed transportation infrastructure for the regions in the SpaceNet 7 dataset. Additionally, we incorporated elevation data from the FAMDEM dataset, allowing us to account for topographical influences on urban growth. The genetic algorithm is still NSGA-II, with the same parameters. We executed one training per dataset. As shown in the Table 4, the only difference in the setup is the deletion of the paths influence because there are no such footpaths in the OpenStreetMap data.

Table 4: Learning variables and constraints for SpaceNet 7 data.

Influence	Parameters	Constraints
Buildings	$\lambda_{\min} = X_0$	$0 \leq X_0 \leq 100$
(Attraction-	$\lambda_0 = X_0 + X_1$	$0 \leq X_1 \leq 100$
Repulsion)	$\lambda_{\max} = X_0 + X_1 + X_2$	$0 \leq X_2 \leq 100$
	$w = X_3/(X_3 + X_7 + X_{10})$	$0 \leq X_3 \leq 1$
Roads	$\lambda_{\min} = X_4$	$0 \leq X_4 \leq 100$
(Attraction-	$\lambda_0 = X_4 + X_5$	$0 \leq X_5 \leq 100$
Repulsion)	$\lambda_{\max} = X_4 + X_5 + X_6$	$0 \leq X_6 \leq 100$
	$w = X_7/(X_3 + X_7 + X_{10})$	$0 \leq X_7 \leq 1$
Slope	$\lambda_{\min} = X_{11}$	$0 \leq X_8 \leq \pi/2$
(Open	$\lambda_{\max} = X_{12}$	$0 \leq X_9 \leq \pi/2$
Distance)	$w = X_{10}/(X_3 + X_7 + X_{10})$	$0 \leq X_{10} \leq 1$
		$X_9 \leq X_{10}$

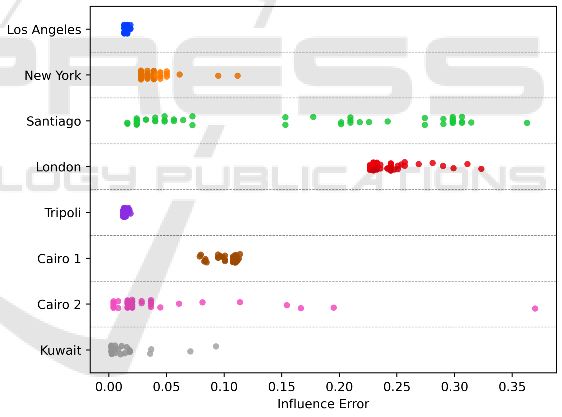

Figure 13: Influence Error SpaceNet 7.

For each city, 10 training runs were conducted, and we then calculated the influence error and simulation error on the resulting influence models. Figure 13 illustrates the influence error for each solution obtained from the genetic algorithm. As shown by this figure, the results of our approach on these datasets are of the same order of magnitude as those obtained for Suva. The simulation error (Figure 14) yields better results. In fact, the influence error is sensitive to the quality of the field data. For example, if the road network is not spatially accurate enough, roads can sometimes "run over" buildings in the data. In this case, the influence model can model that there should be no buildings on the roads, but since this happens in the field data, it will be counted as an error of the influence model. For example, this happens relatively often in the London dataset, which is why performances are worse.

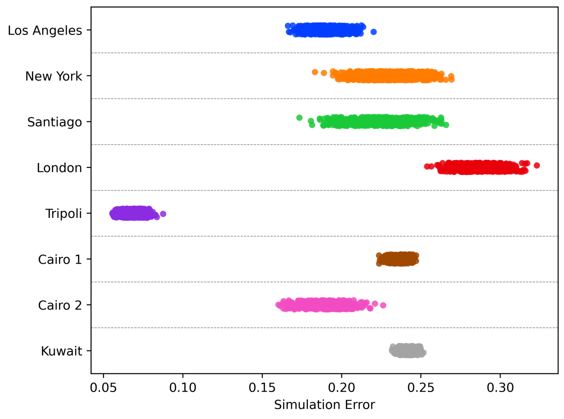

Figure 14: Simulation error for SpaceNet 7.

Figure 15 illustrates the relationship between execution time per generation using one thread (in hours) and the number of new buildings (urban growth) of the different SpaceNet's datasets. Naturally, the more new buildings there are, the longer the simulation time will be. For example, the Kuwait dataset is the bigger, and it is associated to the longest execution time (around 5 hours). In contrast, datasets like New York, London, and Santiago, which have fewer buildings, experience much shorter execution times (under 1 hour).

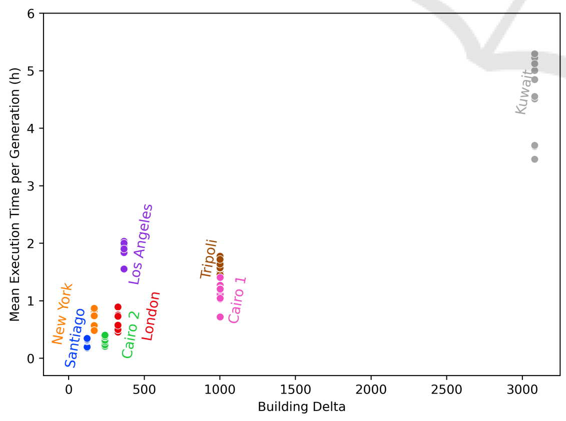

Figure 15: Execution time in relation to the size of the building delta for each dataset.

6 CONCLUSIONS AND PERSPECTIVES

This study introduces a novel approach combining procedural generation and genetic algorithms to model spatial influences in agent-based urban growth simulations. By formalizing environmental factors through mathematical functions and optimizing their parameters with NSGA-II, we generate realistic urban growth patterns, validated using spatial measures like grid-based density differences and Chamfer distance.

However, some notable differences between the simulation results and the actual spatial distribution of buildings remain. Experts believe that this is certainly due to the fact that the human factor has not yet been taken into account in the agent-based model. It is known in the literature that it has a strong impact on the urban growth of such informal settlements. Despite these discrepancies with real-world data, our results demonstrate the method's effectiveness and potential for broader applications.

Future work will include applying the approach to diverse regions, integrating time-series analysis for dynamic modelling, and incorporating socio-economic factors to better reflect human behaviours. Enhancements such as generating realistic building shapes and simulating road and path evolution are also planned, aiming for a comprehensive, adaptable urban modelling framework.

ACKNOWLEDGEMENTS

We would like to acknowledge the French government having supported financially a CIFRE doctoral contract in collaboration with the INSIGHT company. We also thank Thomas Gaillard (Ecosophy) and the University of the South-Pacific for the data acquisition on the site of Valenicina in Fiji, including the Fijian students who participated during the socio-economical survey. And more specifically, Thomas Gaillard for his several expert returns during the modelling.

REFERENCES

Agyemang, F. S. K., Silva, E., and Fox, S. (2022). Modelling and simulating ‘informal urbanization’: An integrated agent-based and cellular automata model of urban residential growth in Ghana. *Environment and Planning B: Urban Analytics and City Science*.

Augustijn-Beckers, E.-W., Flacke, J., and Retsios, B. (2011). Simulating informal settlement growth in Dar es Salaam, Tanzania: An agent-based housing model. *Computers, Environment and Urban Systems*, 35(2):93–103.

Barros, J. (2004). Simulating Urban Dynamics in Latin American Cities. In Atkinson, P., Wu, F., Foody, G., and Darby, S., editors, *GeoDynamics*, pages 313–328. CRC Press.

Blank, J. and Deb, K. (2020). pymoo: Multi-objective optimization in python. *IEEE Access*, 8:89497–89509.

Crooks, A. (2018). *Agent-Based Modelling and Geographical Information Systems: A Practical Primer*. Spatial Analysis and GIS. SAGE Publications, Thousand Oaks, CA, 1st edition edition.

Deb, K., Pratap, A., Agarwal, S., and Meyarivan, T. (2002). A fast and elitist multiobjective genetic algorithm: Nsga-ii. *IEEE transactions on evolutionary computation*, 6(2):182–197.

Emilien, A., Bernhardt, A., Peytavie, A., Cani, M.-P., and Galin, E. (2012). Procedural generation of villages on arbitrary terrains. *The Visual Computer*, 28(6-8):809–818.

Etten, A. V., Hogan, D., Martinez-Manso, J., Shermeyer, J., Weir, N., and Lewis, R. (2021). The multi-temporal urban development spacenet dataset.

Jokar Arsanjani, J., Helbich, M., and de Noronha Vaz, E. (2013). Spatiotemporal simulation of urban growth patterns using agent-based modeling: The case of Tehran. *Cities*, 32:33–42.

OpenStreetMap contributors (2017). Planet dump retrieved from https://planet.osm.org . https://www.openstreetmap.org.

Picascia, S. and Yorke-Smith, N. (2017). Towards an agent-based simulation of housing in urban beirut. In *Agent Based Modelling of Urban Systems*, pages 3–20. Springer International Publishing.

Schelling, T. C. (1971). Dynamic models of segregation. *Journal of mathematical sociology*, 1(2):143–186.

Schwarz, N., Flacke, J., and Sliuzas, R. (2016). Modelling the impacts of urban upgrading on population dynamics. *Environmental Modelling & Software*, 78:150–162.

Smelik, R. M., Tutenel, T., Bidarra, R., and Benes, B. (2014). A survey on procedural modelling for virtual worlds. In *Computer Graphics Forum*, volume 33, pages 31–50. Wiley Online Library.

Tobler, W. R. (1970). A computer movie simulating urban growth in the detroit region. *Economic geography*, 46(sup1):234–240.

Zhang, H., Zeng, Y., Bian, L., and Yu, X. (2010). Modelling urban expansion using a multi agent-based model in the city of changsha. *Journal of Geographical Sciences*, 20:540–556.