

**DEEP REINFORCEMENT LEARNING FOR URBAN  
MODELING: MORPHOGENESIS SIMULATION OF SELF-  
ORGANIZED SETTLEMENTS**



A THESIS SUBMITTED TO  
THE GRADUATE SCHOOL OF ENGINEERING AND SCIENCE  
OF BILKENT UNIVERSITY  
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR  
THE DEGREE OF MASTER OF SCIENCE  
IN  
ARCHITECTURE

By  
Houssame Eddine H'sain  
July 2023

DEEP REINFORCEMENT LEARNING FOR URBAN MODELING:  
MORPHOGENESIS SIMULATION OF SELF-ORGANIZED  
SETTLEMENTS

By Houssame Eddine H'sain

July 2023

We certify that we have read this thesis and that in our opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

---

Burcu Şenyapılı Özcan (Advisor)

---

Yiğit Acar (Co-Advisor)

---

Olgu Çalışkan

---

Başak Uçar

Approved for the Graduate School of Engineering and Science:

---

Orhan Arıkan  
Director of the Graduate School

# ABSTRACT

## DEEP REINFORCEMENT LEARNING FOR URBAN MODELING: MORPHOGENESIS SIMULATION OF SELF- ORGANIZED SETTLEMENTS

Houssame Eddine H'sain

M.S. in Architecture

Advisor: Burcu Şenyapılı Özcan

Co-advisor: Yiğit Acar

July 2023, 117 Pages

Self-organized modes of urban growth could result in high-quality urban space and have notable benefits such as providing affordable housing and wider access to economic opportunities within cities. Modeling this non-linear, complex, and dynamic sequential urban aggregation process requires adaptive sequential decision-making. In this study, a deep reinforcement learning (DRL) approach is proposed to automatically learn these adaptive decision policies to generate self-organized settlements that maximize a certain performance objective. A framework to formulate the self-organized settlement morphogenesis problem as single-agent reinforcement learning (RL) environment is presented. This framework is then verified by developing three environments based on two cellular automata urban growth models and training RL agents using the Deep Q-learning (DQN) and Proximal Policy Optimization (PPO) algorithms to learn sequential urban aggregation policies that maximize performance metrics within those environments.

The agents consistently learn to sequentially grow the settlements while adapting their morphology to maximize performance, maintain right-of-way, and adapt to topographic constraints. The method proposed in this study can be used not only to model self-organized settlement growth based on preset performance objectives but also could be generalized to solve various single-agent sequential decision-making generative design problems.

**Keywords:** self-organized settlements, urban morphology, urban modeling and simulation, machine learning, deep reinforcement learning, agent-based modeling, design control and planning

# ÖZET

## DERİN PEKİŞTİRMELİ ÖĞRENME İLE KENTSEL MODELLEME: KENDİLİĞİNDEN ÖRGÜTLENEN YERLEŞİMLERİN MORFOGENEZ SİMÜLASYONU

Houssame Eddine H'sain

Yüksek Lisans, Mimarlık Bölümü

Tez Danışmanı: Burcu Şenyapılı Özcan

Es Danışman: Yiğit Acar

Temmuz 2023, 117 Sayfa

Kendiliğinden örgütlenen kentsel büyüme modları, yüksek kaliteli kentsel alanlar üretme, uygun fiyatlı konut sağlama ve şehirler içinde ekonomik fırsatlara daha geniş erişim sağlama gibi potansiyelleri barındırır. Bu tip yerleşimler, ve tabandan yukarı örgütlenen tasarım süreçleri kısıtlı alanda ve özellikle eğimli arazilerde, çevresel etmenler açısından da verimli çözümler sunar.

Bu tip doğrusal olmayan, karmaşık ve dinamik sıralı kentsel toplama süreçlerinin modellenmesi, uyarlanabilir, sıralı kararlar almayı gerektirir. Bu çalışmada, belirli performans hedeflerini en üst düzeye çıkaran kendiliğinden örgütlenmiş yerleşimler oluşturmak için derin pekiştirmeli öğrenme (Deep Reinforcement Learning: DRL) yaklaşımı önerilir. Bu yaklaşım kapsamında, kendiliğinden örgütlenen yerleşimlerin morfogenez problemini simüle edecek tekil ajanlara dayalı (single-agent) bir pekiştirmeli öğrenme (Reinforcement Learning:

RL) çerçevesi önerilmiştir. Önerilen çerçeve, iki hücreli otomata temelli kentsel büyüme modeline dayanan üç ortam geliştirilerek, derin Q-öğrenme (DQN) ve Proksimal Politika Optimizasyonu (Proximal Policy Optimization: PPO) algoritmaları kullanılarak, RL ajanlarının eğitilmesiyle ve bu ortamlardaki performans metriklerine dayalı sıralı kentsel oluşum politikaları üretilmesi yoluyla test edilir.

Ajanlar, performansı en üst düzeye çıkarmak, geçiş hakkını korumak ve topografik kısıtlamalara uyum sağlamak için kentsel biçimi uyarlarlarken, yerleşimleri sırayla büyütmeyi sürekli olarak öğrenirler. Bu çalışmada önerilen yöntem sadece önceden belirlenmiş performans hedeflerine dayalı olarak kendi kendine organize edilen yerleşim büyümesini modellemek için değil, aynı zamanda benzer nitelikli tasarım problemlerinin formülizasyonu için de kullanılabilir bir çerçeve sunar.

**Anahtar sözcükler:** kendiliğinden örgütlenen yerleşimler, kentsel morfoloji, kentsel modelleme ve simülasyon, makine öğrenimi, derin pekiştirmeli öğrenme, ajan tabanlı modelleme, tasarım kontrolü ve planlama

## ACKNOWLEDGEMENTS

I am deeply grateful to my advisors, Assoc. Prof. Burcu Şenyapılı Özcan Ph.D. and Yiğit Acar Ph.D. for their unwavering support and guidance throughout my master's program. Their expertise and patience have been invaluable to me and have played a crucial role in the success of this thesis. They have helped me grow intellectually and guided me towards a better understanding of urban planning, computational design, among many other topics, and instructed me in how to conduct scientific research.

I am grateful to the Bilkent University Department of Architecture (BUDA) for providing me with the opportunity to conduct my research and for all the resources and support they provided throughout the years. I would like to extend a special thanks to all the faculty members and instructors that I had the privilege to learn from while attending this school of architecture during both my undergraduate and graduate studies.

I would also like to thank Assoc. Prof. Olgu Çalışkan Ph.D. and Assist. Prof. Başak Uçar Ph.D. for serving on my thesis committee and providing valuable feedback and suggestions. Their insights and guidance were instrumental in helping me to shape my research and improve this thesis further. They also provided me with valuable insight into future research directions related to my thesis topic.

I am deeply thankful to my family and friends for their love, support, encouragement, and motivation during this process. Their presence made the hardest moments much easier and easiest moments far more joyful and enjoyable. I would not have been able to complete this journey without their unconditional support.

Finally, Zeina, thank you for existing.

# TABLE OF CONTENTS

|                                                                                                                                     |             |
|-------------------------------------------------------------------------------------------------------------------------------------|-------------|
| <b>ABSTRACT .....</b>                                                                                                               | <b>iii</b>  |
| <b>ÖZET .....</b>                                                                                                                   | <b>v</b>    |
| <b>ACKNOWLEDGEMENTS .....</b>                                                                                                       | <b>vii</b>  |
| <b>TABLE OF CONTENTS.....</b>                                                                                                       | <b>viii</b> |
| <b>LIST OF FIGURES.....</b>                                                                                                         | <b>x</b>    |
| <b>LIST OF TABLES .....</b>                                                                                                         | <b>xiv</b>  |
| <b>LIST OF ABBREVIATIONS .....</b>                                                                                                  | <b>xv</b>   |
| <b>1 Introduction.....</b>                                                                                                          | <b>1</b>    |
| 1.1 Research Problem Statement.....                                                                                                 | 2           |
| 1.2 Aim of the study and research questions.....                                                                                    | 4           |
| 1.3 Methodology of the research.....                                                                                                | 5           |
| 1.4 Structure of the thesis .....                                                                                                   | 6           |
| <b>2 The Study of Self-Organized Settlements .....</b>                                                                              | <b>8</b>    |
| 2.1 Self-organized settlements: A theoretical perspective .....                                                                     | 8           |
| 2.1.1 Traditional settlements' morphogenesis.....                                                                                   | 10          |
| 2.1.2 Informal settlements as current instances of self-organized settlements.....                                                  | 14          |
| 2.1.3 Self-organized settlements as complex adaptive systems .....                                                                  | 18          |
| 2.2 Digital modelling of self-organized settlement morphogenesis .....                                                              | 23          |
| 2.2.1 The history of digital modelling methods in urban design .....                                                                | 24          |
| 2.2.2 Attempts of self-organized settlement morphogenesis digital modelling.....                                                    | 27          |
| 2.2.3 Limitations of rule-based methods and the potential of learning-based<br>methods in settlement morphogenesis simulation ..... | 33          |
| <b>3 Reinforcement Learning in Design .....</b>                                                                                     | <b>35</b>   |
| 3.1 Reinforcement learning fundamentals .....                                                                                       | 35          |
| 3.1.1 Artificial intelligence, machine learning, and deep learning .....                                                            | 36          |
| 3.1.2 The reinforcement learning framework .....                                                                                    | 41          |
| 3.1.3 Markov decision processes and reinforcement learning .....                                                                    | 44          |
| 3.2 The Markov decision process for generative design problems .....                                                                | 46          |
| 3.3 Reinforcement learning applications in design .....                                                                             | 49          |

|                                                                                |           |
|--------------------------------------------------------------------------------|-----------|
| <b>4 Experiments and Results .....</b>                                         | <b>53</b> |
| 4.1 Self-organized settlement morphogenesis as a Markov decision process ..... | 53        |
| 4.2 Experiment Setting .....                                                   | 58        |
| 4.2.1 The Generalized Beady Ring Environment version 1 .....                   | 58        |
| 4.2.2 The Generalized Beady Ring Environment version 2 .....                   | 60        |
| 4.2.3 The Generalized Path Node Environment .....                              | 61        |
| 4.3 Results and Discussion .....                                               | 63        |
| <b>5 Conclusion .....</b>                                                      | <b>74</b> |
| 5.1 General evaluation .....                                                   | 74        |
| 5.2 Limitations of the method .....                                            | 75        |
| 5.3 Potential applications of the method.....                                  | 77        |
| <b>Bibliography .....</b>                                                      | <b>78</b> |
| <b>Appendix.....</b>                                                           | <b>93</b> |

## LIST OF FIGURES

Figure 2.1 Determining the height of windows for the preservation of privacy in Islamic cities. The above sketch shows how to determine the height from the interior if the opening overlooks the private domain of the neighbors. The lower two sketches indicate how to determine the height of the windowsills from the street to prevent looking into the interior. (Reprinted from Hakim, 1986, p.34)

Figure 2.2 Informal building rights. Diagram by Kim Dovey. (Reprinted from Dovey et al., 2020, p.13)

Figure 2.3 CAS scenario visualized from the point of view of a single actor within a 4-actor simulation environment. The actor performs actions within the environment based on feedback received from the environment representing the effects of actions performed by other actors as well as natural constraints within that same environment. Produced by author.

Figure 2.4 Sample output of settlement simulation generated by the 'Path Node Generator' model. (Reprinted from Erickson and Lloyd-Jones, 1997, p.919).

Figure 2.5 Rule-based urban settlement modeling approach as opposed to the proposed learning-based urban modeling approach. Produced by author.

Figure 3.1 The relationship between artificial intelligence (AI), machine learning (ML), and deep learning (DL) and their different subdomains as well as the different categories of tasks they are most suitable for. Produced by author.

Figure 3.2 The three fundamental frameworks of machine learning. (Reprinted from Wang and Snooks, 2021, p.191)

Figure 3.3 The agent-environment interaction in reinforcement learning. Produced by author based on Sutton & Barto, 2015.

Figure 3.4 A simple MDP example with three possible states and two allowable actions,

with transition probabilities from one state to another, and resulting reward signals given the current state and the action chosen by the agent. Produced by author.

Figure 3.5 The generative design problem formulated as an MDP. The MDP could be solved according to a preset objective using various DRL methods. Produced by author.

Figure 3.6 Sample of the sequential generation process of an urban block by a DRL agent. (Reprinted from Han et al., 2021, p.141)

Figure 4.1 SOS-MDP framework diagram. All the necessary components to define an MDP are defined for the self-organized settlement case. The MDP could be solved according to a preset objective using various DRL methods. Produced by author.

Figure 4.2 The Moore neighborhood (a) and the von Neumann neighborhood (b). Produced by author.

Figure 4.3 The GBR environment with local observation, represented by the green area. Produced by author.

Figure 4.4 A cellular automata grid (a) and its labeled clusters using the Hoshen-Kopelman algorithm. In this example, the agent receives a reward only for cluster number 2, since it has six cells. Produced by author.

Figure 4.5 A sample topography generated using the Perlin Simplex noise algorithm, with slope values normalized between 0 and 1. Produced by author.

Figure 4.6 Training results on the GBR version 1 environment using the DQN and PPO algorithms. Produced by author.

Figure 4.7 Renders of the terminal states of the training episodes of the GBR version 1 environment using both the DQN and PPO algorithms. The renders are taken every 50 episodes with 32 parallel processes during training. Produced by author.

Figure 4.8 Training results on the GBR version 1 environment using the DQN algorithm with both global and local observation. Produced by author.

Figure 4.9 Training results on the GBR version 2 environment using the PPO algorithm. Produced by author.

Figure 4.10 Renders of the terminal states of the training episodes from 2 parallel instances of the GBR version 2 environment using PPO algorithm. The renders are taken every 50 episodes with 32 parallel processes during training. Produced by author.

Figure 4.11 Sample settlement configurations generated using PPO within both versions of the GBR environment, showcasing the effect of the reward signal on the policy learned by the agent. Produced by author.

Figure 4.12 Renders of the terminal states of the training episodes of the GPN environment using PPO. The renders are taken every 50 episodes with 32 parallel processes during training. Produced by author.

Figure 4.13 Training results on the GPN environment using the DQN and PPO algorithms. Produced by author.

Figure 4.14 Renders of the terminal states of the training episodes of the GPN environment with different topographic constraints (introvert, linear, and extrovert) using PPO. Produced by author.

Figure 4.15 The process of training and then deploying trained agents on similar urban growth simulation environments. Produced by author.

Figure a.1 Sample settlement representation output within the GBR v3 environment. The street network is represented in blue segments. Produced by author.

Figure a.2 Preview of settlement generated within an environment implemented in Rhino/Grasshopper. Produced by author.

Figure a.3 HoopSnake component environment implementation variant. Produced by author.

Figure a.4 GhPython component environment implementation variant. Produced by author.



## LIST OF TABLES

Table a.1 Hyperparameters used to train DQN agents in each environment. Moore refers to the local observation version of the GBR v1, while CNN means Convolutional neural network referring to the global observation version of that same environment. Produced by author.

Table a.2 Hyperparameters used to train PPO agents in each environment. Produced by author.

Table a.3 Perlin Simplex noise parameters used to generate the topographic conditions within the GPN environment. Produced by author.

## LIST OF ABBREVIATIONS

| Abbreviation | Definition                           |
|--------------|--------------------------------------|
| A2C          | Advantage actor critic               |
| ABM          | Agent-based modeling                 |
| AI           | Artificial intelligence              |
| AIAD         | Artificial intelligence-aided design |
| ANN          | Artificial neural network            |
| CA           | Cellular Automata                    |
| CAD          | Computer-aided design                |
| CAS          | Complex adaptive system              |
| CNN          | Convolutional neural network         |
| DCNN         | Dynamic convolutional neural network |
| DDPG         | Deep deterministic policy gradient   |
| DDQN         | Double deep Q-network                |
| DL           | Deep learning                        |
| DNN          | Deep neural network                  |
| DQN          | Deep Q-network                       |

|         |                                                   |
|---------|---------------------------------------------------|
| DRL     | Deep reinforcement learning                       |
| GAN     | Generative adversarial network                    |
| GBR     | Generalized Beady Ring                            |
| GIS     | Geographic information system                     |
| GPN     | Generalized Path Node                             |
| GPU     | Graphics processing unit                          |
| HPC     | High performance computing                        |
| MADRL   | Multi-agent deep reinforcement learning           |
| MAS     | Multi-agent simulation                            |
| MDP     | Markov decision process                           |
| ML      | Machine learning                                  |
| PPO     | Proximal policy optimization                      |
| RL      | Reinforcement learning                            |
| SAC     | Soft actor critic                                 |
| SOS-MDP | Self-organized settlement Markov decision process |
| TD3     | Twin-delayed deep deterministic policy gradient   |
| WFC     | Wave function collapse                            |

# CHAPTER 1

## 1 Introduction

The availability of data and the relative low-cost of acquiring digital forms of information nowadays, coupled with the accessibility to powerful computation resources in the form of graphics processing units (GPUs) resulted in a surge of artificial intelligence (AI) applications in different scientific domains including architecture and urban design. AI, and more specifically machine learning (ML), offer a shift in paradigm in design thinking and have been proven to be effective in tackling some complex design and planning challenges (Huang and Zheng, 2018; Kelly et al., 2018; Nauata et al., 2020; Uzun et al., 2020; Wang et al. 2020; Veloso and Krishnamurti, 2020; Zhang and Blasetti, 2020; Han et al., 2021; Zheng and Yuan, 2021; Ororbia & Warn, 2022).

One such challenging design problem is self-organized settlement morphogenesis modeling, as it is complex, dynamic and ever-evolving (Batty, 1971; Allen, 1997); requiring a non-linear adaptive model to simulate and control. Interest in urban morphology and its impact on environmental and socio-economic conditions has increased in recent decades, especially in research areas targeting new self-organized and informal urban settlements in emerging economies (Rapoport, 1977; Polidoro et al., 2012; Leach, 2015; Mobaraki & Oktay Vehbi, 2022). Self-organized settlements are generated in a bottom-up manner, which makes the concept a viable test case for reinforcement learning (RL), a subdomain of machine learning. RL aims to train agents to learn adaptive behaviour through iterative cycles of trial and error. RL methods

entered the computational design field very recently, with applications in form finding, structural optimization, and urban design to name a few. RL is considered a new paradigm in computational generative design (Wang & Snooks, 2021) and RL techniques gained popularity in the last decade due to their effectiveness in solving some of the most complex board and video games. These techniques are also used successfully in industry today in domains such as robotics and autonomous systems. This thesis aims at investigating reinforcement learning, more specifically deep reinforcement learning (DRL) techniques applied to the modelling and simulating self-organized settlements and proposes a generalizable DRL framework to simulate any self-organized settlement's morphogenesis.

## **1.1 Research Problem Statement**

Self-organized settlements constitute a considerable portion of urban fabric globally today. Increased levels of attention should be given to research in methods that would enable better design and production of these urban spaces. An ever-increasing number of studies introduced performance-based approaches to formal urban space generation considering various performance metrics pertaining to the environmental impacts as well as economic, social, and spatial dimensions (Chang et al., 2019; Chatzipoulka et al., 2016; Chokhachian et al., 2020; Koenig et al., 2020). These efforts provide planners and urban designers with new tools and techniques to ensure the quality and sustainability of proposed developments in a much more efficient manner compared to traditional planning and design practice methods. However, few of these methods focus on self-organized settlement generation, and none of the studies investigate DRL methods for performance-based self-organized settlement generation. This thesis will investigate such methods and propose a system to generate and optimize settlements based on various spatial and environmental metrics.

In an effort to uncover the governing logics of self-organized settlement morphogenesis, researchers studied many informal and traditional settlements around the Mediterranean basin mainly, but also in locations in South America, Sub-Saharan Africa, and South Asia (Hakim & Ahmed, 2006; Duarte et al., 2007; Arefi, 2011;

Hakim, 2014; Porta et al., 2014; Kamalipour, 2016; Iovene et al., 2017; Jones, 2019; Kamalipour & Dovey, 2020). They managed to uncover the fact that there are striking similarities between traditional settlements and modern-day informal settlements in terms of form generation dynamics, rules and morphogenic outcomes. For instance, Iovene et al. (2017) discovered patterns of development in a Lima community by evaluating the urban form over a period of around 70 years, based on the idea that, similar to informal settlements, pre-modern cities change at a fast pace. Porta et al. (2014) discovered comparable patterns in informal and historic street networks, implying that informal settlements might be used as modern instances of historic cities' self-organization logic. Many studies also uncovered that there are general rules that govern the morphogenesis of all self-organized settlements (Forrester, 1970; Batty, 1971; Batty, 1972; Allen & Sanglier, 1981; Allen, 1997; Cecchini et al., 2000; Batty, 2005; Batty, 2008).

Attempts to model the self-organized settlement generative process are found within the literature (Hillier & Hanson, 1984; Erickson & Lloyd-Jones, 1997; Batty et al., 1999; Hillier et al. 2000; Duarte et al., 2007; Koenig & Bauriedel, 2009; Heath et al., 2009; Barros et al., 2013; Dias, 2014; Heppenstall et al., 2016; Verniz & Duarte, 2017; Rahimian et al., 2020). Some of these attempts focus on a specific settlement case study, extracting a set of specific rules that govern the morphogenesis of that specific settlement, and then proposing an algorithmic process to regenerate those same settlements. These methods are effective at replicating the morphogenic process of those specific settlements but fail to capture the general rules that govern the morphogenesis of self-organized settlements, and therefore these models could not readily be used to study other settlements or to simulate the generation of new self-organized settlements. A potential generalizable way to capture and encode these generative rules would be to learn through trial and error. Through the reinforced interaction with a self-organized settlement environment, a learning model would ideally figure out a way to achieve higher reward; a feedback signal that rewards favorable conditions within the generated settlement, and penalizes negative ones, and therefore, encode the general generative rules that result in a satisfactory, well-functioning settlement. The field of RL offers the framework to test this new modelling potential. No study has been found within the

literature testing modern DRL methods on the modelling and simulation of self-organized settlements. This thesis investigates the application of some of the most well-established DRL methods on self-organized settlement morphogenesis modelling and proposes a system to test various algorithms on any settlement simulation environment. This proposed system is then tested on a few environments developed based on some of the well-known self-organized settlement models found within the literature to validate its efficacy, and also identify its limits.

## 1.2 Aim of the study and research questions

The main objective of this thesis is to investigate the deep reinforcement learning framework as a method for self-organized settlement morphogenesis modelling. By doing so, this thesis proposes a system using DRL techniques for modelling a wide variety of self-organized settlements. This system is applied to a set of environments developed based on well-established urban morphology generative models, namely the ‘Beady Ring’ model by Hillier and Hanson (1984), and one of the models proposed by Erickson and Lloyd-Jones (1997). This research aims at demonstrating the potential of DRL methods in modelling self-organized settlement morphogenesis, and also identifying their limitations.

The first research question of this thesis is **‘whether RL is suitable for self-organized settlement morphogenesis modelling.’** To answer this question, this thesis provides a theoretical justification to the pertinence of the use of RL methods in self-organized settlement morphogenesis modelling, and also support it by practical applications of these methods to sample self-organized settlement generative environments. This is a contribution of this thesis, as it lays the theoretical ground for further research and investigation in DRL applications in urban morphogenesis in particular, and in other areas of computational generative design.

The second research question is **‘what should be the structure of an RL system for modelling self-organized settlements?’** This is the part where this research explains in detail all the components of the proposed system. It is crucial to understand the basic components of the RL framework and adapt it to the urban modelling task of

this research by identifying the relevant model architectures and specific RL learning systems that could be used. By answering this question, this thesis provides a way to train DRL agents on various urban generative models that could be employed by future research.

The third and last question of this thesis aims at identifying **‘whether the proposed DRL modelling system could be applied with success to different self-organized settlement generative environments.’** This is important for the applicability of the proposed method to future work, and ultimately to its relevance to real-world self-organized settlement morphogenesis scenarios. The proposed method could only be useful if it has the capacity to be applicable to different settlement modelling environments without a radical change in its structure. The aim here is to provide a method that could train agents that simulate the self-organized morphogenesis of various settlements with only changes occurring on the environment setup and not on the proposed training framework. With that, the proposed method should to some extent provide successful models for each of those environments. This would be evaluated by training agents on four different environments with various specifications and levels of complexity.

### **1.3 Methodology of the research**

This research selects two of the most prominent DRL algorithms within the RL community: (i) Deep Q-Network (DQN) (Mnih et al., 2013) and (ii) Proximal Policy Optimization (PPO) (Schulman et al., 2017). These algorithms are then applied to three environments developed based on urban generative models proposed by Hillier and Hanson (1984) and Erickson and Lloyd-Jones (1997): (i) the Generalized Beady Ring environment with two different versions, and (ii) the Generalized Path Node environment. The environments range from a simple cellular automata grid world model to a more complex non grid bound cellular automata model that introduces different orientations to cells, therefore enabling more elaborate urban forms. The results obtained by training DRL agents on these environments are discussed and would help in identifying the potential of DRL methods in performance-based self-organised settlement generation. These conclusions could also be generalized to similar generative

urban design tasks.

## 1.4 Structure of the thesis

This thesis is composed of five chapters. **The first and current chapter** serves as an introduction to the research problem under study and declares the aim and objectives of this research. It identifies a research gap within the literature and put forth a methodology to answer the research questions.

**The second chapter** situates the current research within the urban morphology literature. It provides a review of self-organized settlement studies and identifies the current state-of-the-art in self-organized settlement digital simulation and modelling and discussing their limitations. This chapter also clarifies and defines some important theoretical concepts and terminology essential to this research. The purpose of this chapter is to situate this thesis within the research efforts towards a better understanding of self-organized settlements and more precisely elaborating on new digital methods and tools that would enable that better understanding. In this second chapter, major case studies of self-organized settlements within the literature are listed with more attention given to the results and conclusions taken by the authors about self-organized settlements. These conclusions are important in this thesis' attempt at formulating an RL framework for self-organized settlement simulation.

**The third chapter** introduces RL and the fundamental components of its framework. It introduces new terminology related to this machine learning subfield, and therefore lays out the theoretical framework that is important in understanding the current research's proposed method. This chapter also contains a literature review of RL applications in architecture, urban design, and other related fields such as structural optimization and topology optimization.

**The fourth chapter** introduces this thesis' proposed RL method for self-organized settlement morphogenesis modelling. It explains all the different components of the method and the technical details of it. It introduces four different self-organized settlement RL environments with increasing levels of complexity used to test various RL algorithms. It also contains the results of the training of these various RL agents. This

chapter also shows how the proposed method could be implemented and used on different self-organized settlement cases.

**The fifth and last chapter** includes a discussion of the results of the thesis, showcasing the utility of the proposed method in modelling self-organized settlements as well as its limitations. The chapter also offers routes for further studies, aimed at improving on the performance of the proposed method as well as investigating other related methods not covered within the scope of this thesis.



## **CHAPTER 2**

### **2 The Study of Self-Organized Settlements**

#### **2.1 Self-organized settlements: A theoretical perspective**

Urban morphology is the study of the physical form of settlements. It aims at analysing urban fabric components and the relationship between them to describe their composition and configuration through time. This enables the prediction and identification of the urban form's spatial socio-economic impacts, as well as the environmental impact of urban settlements. Under this domain of study, there are two distinct sets that seek to answer different questions that pertain to the morphological aspect of urban settlement. The first set focuses on explaining and describing the current form of a settlement (Dovey et al., 2012, 2016, 2020; Gouverneur, 2015; Hakim, 2014; Kamalipour, 2018, 2021), and the second seeks to predict and generate new forms of human settlement in an effort to identify suitable and better functioning types of morphology (Hillier and Hanson, 1984; Batty, 2005; Duarte et al. 2007; Barros et al., 2013; Dias, 2014; Verniz and Duarte, 2017; Koenig et al., 2020). This thesis uses the knowledge provided by studies on current settlements of the first set and introduces a method of modelling and generating a specific type of settlements called 'self-organized settlements' as in the latter set.

Self-organized settlement is the initial form of human settlements before there was a need for formal urban design and intervention. Self-organization gave rise to some of the most interesting traditional settlements in the world. These settlements happen to offer well-functioning urban spaces that incorporate the same characteristics that mark the most vibrant mixed-use, socially, and formally diverse inner-city neighborhoods of many formal cities (Dovey, 2019). The term self-organized settlement refers to any urban fabric generated through the interaction and cooperation between various agents in a specific urban space. These agents could be individuals, households, private or government agencies. The self-organization characteristic resulting from the sequential decision making at various scales of intervention of these agents makes the settlement a complex dynamic system (Daffertshofer et al., 2001). Viewing self-organized settlements from the lens of complex adaptive systems helped researchers make progress in understanding and modeling such settlements. In her seminal work on urban design Jacobs (1961) framed the city as a system of ‘organized complexity’ displaying dynamic and evolutionary behavior with an implicit morphogenesis. Recent studies of self-organized settlement analysis in various parts of the world (Dovey et al., 2012, 2016, 2020; Gouverneur, 2015; Hakim, 2014; Kamalipour, 2018, 2021) all concur on the complex adaptive system characteristic being common to all the studied settlements and being the prime mechanism to explain their formation and to even predict their future growth.

Continuous research efforts towards uncovering the morphogenic rules of self-organized settlements feature within literature. The understanding of self-organized settlements’ morphogenesis is still relatively limited; therefore, this is an important area of investigation. One of the most notable and insightful works within the literature are perhaps the work of Hakim (2014) on the generative codes that shaped traditional Mediterranean settlements and the works of Dovey (2012, 2018, 2019, 2020) and Kamalipour (2018, 2021) on the morphogenesis of informal settlements, portraying this process as a mode of production of urban space that can lead to favorable urban conditions, and how it relates and interacts with the formal urban fabric.

Previous research established that there exists a set of general generative rules common to all self-organized settlements (Batty, 1971; Allen, 1997; Cecchini et al., 2000; Dovey, 2012). Informal settlements are an instance of self-organized settlements. They are not hierarchically organized like formal settlements; their morphogenic process is based on the constraints of the environmental context and the needs of the community. They are generated according to a procedural bottom-up approach rather than a top-down one (Allen, 1997). While informal morphologies vary, the most informal of settlements have a similar morphogenic process that results in a generally constant pattern of access networks that is visible in many quite distinct cities in various climates and topographies (Dovey, 2019). Each settlement emerges within a particular context (politics, culture, climate, topography) to produce a particular set of urban outcomes in terms of built form and public access networks. Numerous studies reverse-analyzed the morphogenic process of self-organized settlements to find the rules and patterns that explain the current state of these settlements and uncover the ways in which such settlements can become dysfunctional.

The current research takes interest in studies of traditional settlements and informal settlements, as instances of urban settlement generated following a self-organized process, to uncover the characteristics of this process and some of its general rules and dynamics that researchers were able to distill from studying various settlements in different contexts. The morphologic processes that emerge from a specific geographic or cultural context do not fall under the scope of this research. These change from one settlement to another. This thesis focuses on the general dynamics that are common to all settlements, independent of their location or context.

### **2.1.1 Traditional settlements' morphogenesis**

Traditional settlements worldwide have been the focus of research into their urban morphology and generative processes. This is due to the fact that traditional settlements usually provide adequate urban spaces with positive qualities such as **compactness, human scale, flexibility and adaptability over time** to the inhabitants' socio-economic needs and also to the constraints related to the natural environment. These settlements offer a unique glimpse into how humans have interacted with their

environment throughout history. Traditional settlement patterns worldwide have been generally categorized by their morphogenesis or how they have physically developed over time. The most notable works of analysis of traditional urbanism and its morphogenesis found within the literature are the works of Besim Hakim (Hakim, 1986; Hakim, 2008; Hakim, 2014; Hakim & Ahmed, 2006). These studies, spanning centuries of traditional urban development, analyzed traditional Mediterranean urbanism as well as settlements in Sub-Saharan Africa in search of a better understanding of the rules and codes that controlled the emergence of such settlements.

Since the start of human settlement in communities, mainly due to the advent of agriculture, individuals and societies sought to develop and respect a particular set of rules to provide individual needs in terms of dwelling and commerce as well as establish a specific organization and hierarchy of the community reflected on the morphology of these settlements. This organization mechanism involved many actors at different scales, from the individual inhabitants and local authorities to state norms and laws that marked and directed the morphogenesis of the settlements of the most advanced civilizations of the time in terms of urbanism. Both Byzantine and Arabic Islamic codes, which directed the emergence of settlements around the Mediterranean basin from the 5th century to the early 20th century, have plenty of overlapping similarities due to their common goals and intentions towards managing land use and adapting settlements to fit better the economic and social norms and customs of those localities (Hakim, 2008). These codes aimed to organize the adaptation of the built environment to change and grow with minimum damage to various rights by maintaining some level of fairness and equilibrium between all the involved parties within that built environment. Any intervention by an individual or a group, even though allowed in principle by the right of property and ownership, must respect the preexisting situation of use and the rights emanating from that within the neighboring area. These private interventions were in no case allowed to infringe on the public space and deteriorate it in any way. These principles existed even earlier within the *Astynomoi* law of Pergamon, which ruled

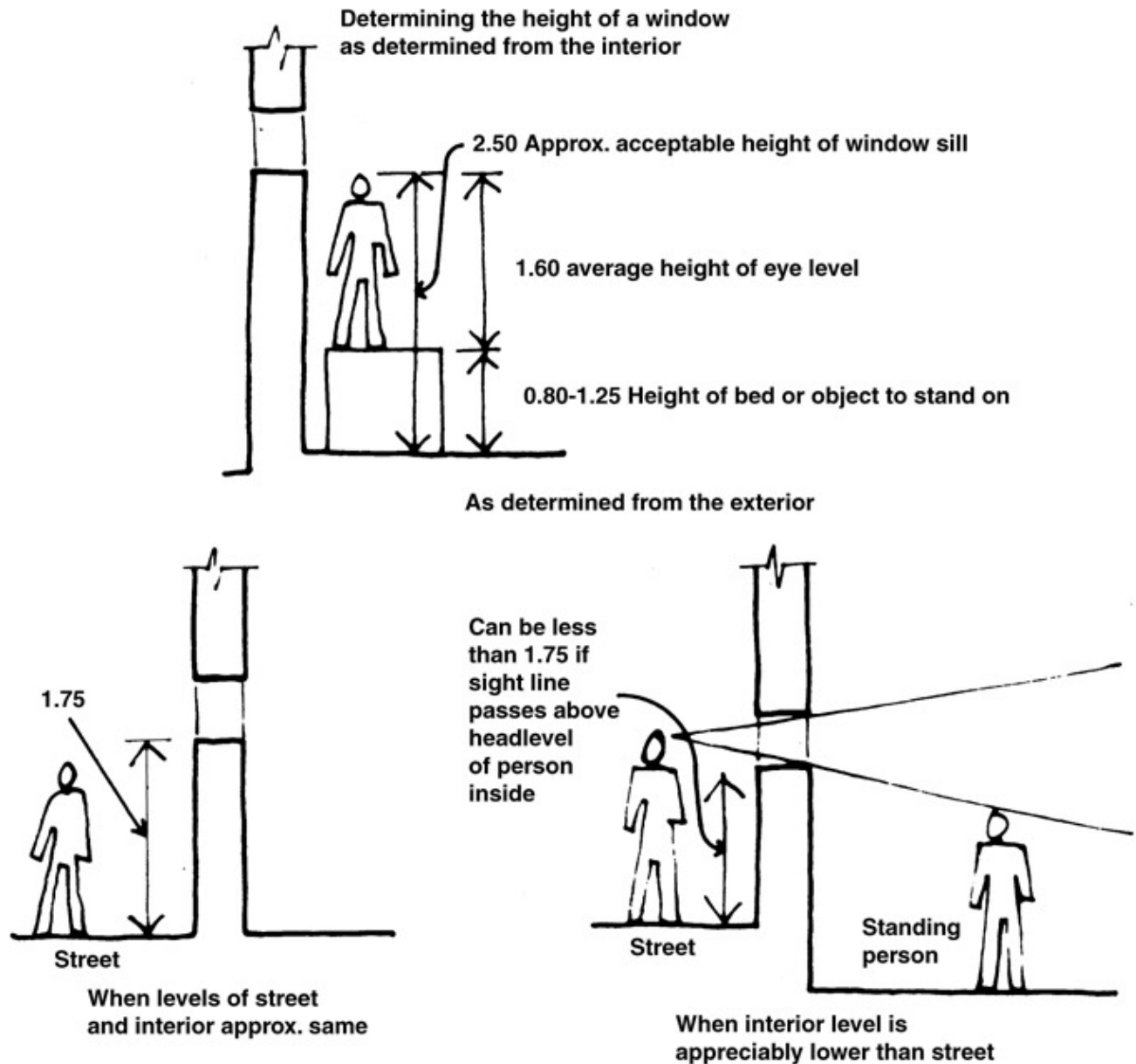


Figure 2.1 Determining the height of windows for the preservation of privacy in Islamic cities. The above sketch shows how to determine the height from the interior if the opening overlooks the private domain of the neighbors. The lower two sketches indicate how to determine the height of the windowsills from the street to prevent looking into the interior. (Reprinted from Hakim, 1986, p.34)

Greek built environment development under Roman rule around the second century (Hakim, 2008). This law stated that public spaces must be protected and maintained regularly, and each person had to repair and clean the public street outside their own house to ensure that public rights of access were not blocked. The same applied to owners of commercial and industrial facilities who were fined if those activities spilled

onto the public street rendering it either dirty or impassable (Watson, 1985, as cited in Hakim, 2014).

Traditional urbanism paid close attention to preserving pleasant views considered as assets to any overlooking property. Byzantine and Greek codes accorded a great deal of importance to views towards the sea, mountains, gardens, orchards, and various landmarks. This was particularly evident in the settlements of the Roman period as well as the late 5th century Constantinople (Hakim, 2008). Visual corridors were also carefully designed and adapted during the accretion process of a settlement to maintain visual privacy from the public space into the private realm (Figure 2.1). This was particularly true for settlements that emerged under Islamic rule in North Africa, Spain, and southern Italy, where pleasant views were a secondary concern relative to obstructing visual connections into private property (Hakim, 1986). For instance, towns developed during the 10th to the 12th centuries in the southern regions of Italy, which followed a set of codes known as '*Prochiron Legum*' that could be traced back all the way to Byzantine Emperor Justinian's Civil Code and had influences from Islamic urbanism, reflected the principles of protection of visual privacy and prominent valuable views in their morphogenesis. Neighbors were not allowed to open windows that would hinder the privacy of a preexisting neighboring private space. Access to sea views was also protected in those settlements. The sea views of a property could not be obstructed by the actions of the owner of any of the neighboring seaward buildings within a radius of around 30 meters (Hakim, 2014).

The customary laws and codes influencing the morphogenesis of traditional settlements were a result of consensus and adaptation to the needs and rights of various owners and inhabitants of those settlements. These codes were enforced by a central authority at times aimed at maintaining the freedom of action of all inhabitants due to their rights of ownership and also at stimulating or penalizing those actions based on their intentions and their expected consequences on neighboring properties and on the public domain in order to prevent any damages to others and encourage favorable interventions that benefit the owner as well as the surrounding built environment as a whole. These morphogenic rules were constant even within settlements that evolved

under a lack of central authority. Within the Greek context, for instance, several towns' self-organized morphogenic process during the period of approximately 1500s to the 1900s has resulted in unique architectural and urban landscape features, still admired by visitors from all parts of the world even though they were produced without prior planning or any rule enforcing central authority (Hakim, 2008).

The generative codes of traditional settlements insist that any intervention must respect the preexisting conditions, especially within the immediate neighboring built environment; however, adaptation, change and growth are encouraged to make sure that the settlement's spatial configuration is adequate in relation to both socio-economic needs as well as environmental constraints. This process of incremental adaptive accretion of settlements is a direct result of a situation where numerous actors make decisions serving their own benefits but under the constraints imposed by the environment and all the competing and conflicting needs and demands of other actors within it in the form of rules, regulations, and customs. This process leads to a form of equilibrium that directly dictates a settlement's form. In decision and game theory, this form of equilibrium is referred to as the Nash equilibrium, named after the American mathematician John Nash. It is a state reached when every single actor takes a set of actions that yields the maximum possible reward given the other actors' actions within the environment (van Damme, 2001). Viewing self-organized settlement morphogenesis from the lens of decision science is therefore rationalized and would hold remarkable potential.

### **2.1.2 Informal settlements as current instances of self-organized settlements**

With accelerating urbanization within the second half of the last century and the beginning of the current century in the developing world, city planning authorities often have a hard time keeping up with this increasing rate of urban growth. Populations move in large numbers from rural areas in search of economic opportunities, higher living standards, and better prospects. A new mode of urban space creation was necessary to keep up with the dwelling and living needs of this incoming population. This mode of urbanization does not follow preset planning rules and regulations but rather evolves over time to adapt to the direct needs of the population and the various constraints

imposed by the natural environment, similar to what used to happen within traditional settlements. It is therefore called an informal mode of generation of urban fabric, a bottom-up approach as opposed to the top-down approach of formal urbanism. Although informal settlements produce accessible land, affordable housing, and urban infrastructure, little is known about how this infrastructure emerges in the first place. While they are commonly perceived as chaotic, informal settlements are a type of urban design and planning that embodies a complex set of poorly understood logics (Dovey et al., 2020).

Informal settlement is defined as the incremental, unauthorized, and self-organized production of new urban neighborhoods (Dovey et al., 2020). The logic of informal settlement is an implicit organization based on rules and patterns that are flexible, adaptable, and difficult to detect (Arefi, 2011). While informal morphologies vary, the most informal settlements have a relatively similar morphogenic process that results in a relatively consistent pattern of access networks that is visible in many very different cities, climates, and topographies (Dovey, 2019). In terms of built form and public access networks, each settlement develops within a specific context and a set of constraints emanating from the community's politics, culture, and socio-economic circumstance, as well as climate and topographic conditions (Dovey et al., 2020). Informal morphogenesis is strongly influenced by natural topography as well as existing infrastructure. Informal settlements are frequently located on inaccessible or unstable land, such as a waterfront, marsh, or escarpment. In such cases, informal settlement generally produces new access to previously inaccessible topography. However, the flattest or most stable sites are usually developed first (Dovey et al., 2020). Encroachment and growth often happen on the most accessible and easily buildable land. This is a determining factor of the morphogenesis of informal settlements since it leads to a sequential aggregation of buildings and streets in which growth is restricted by harsh topographic conditions that make building a costlier endeavor, but that same growth is also encouraged by preexisting infrastructure as it facilitates access and building procedures (Kamalipour, 2016). This preexisting infrastructure is mostly found within the fringes of formal urban fabric. The informal settlement grows incrementally starting from favorable building and access conditions laid out previously by formal

development of urban space. Informal settlements grow and expand through small local increments and building renovations such as small additional rooms, roof terraces, balconies, and stairways. However, these encroachments, which usually happen at the expense of adjacent public space, are governed, and regulated by a set of informal rights and codes such as the rights of way, air rights, and the right of light and ventilation (Figure 2.2) (Kamalipour & Dovey, 2020). Kamalipour (2016) reached these same conclusions and reinforced them through an empirical study of the informal Khlong Toei district in Bangkok, Thailand, where swamp land in the area has been encroached through single-room and multi-room additions over time by workers in need for housing. Morphological studies of various cities in the developing world uncovered that the majority of urban fabric is a mixture of self-organized informal development around formal predesigned structuring urban spaces (Dovey & Kamalipour, 2018). These mixed areas include both formal settlements in a process of becoming informalized and informal settlements gradually becoming formalized. The interaction between the informal and formal creates possibilities for complex and rich urban forms similar to the conditions found within traditional settlements. This reciprocity between the formal and

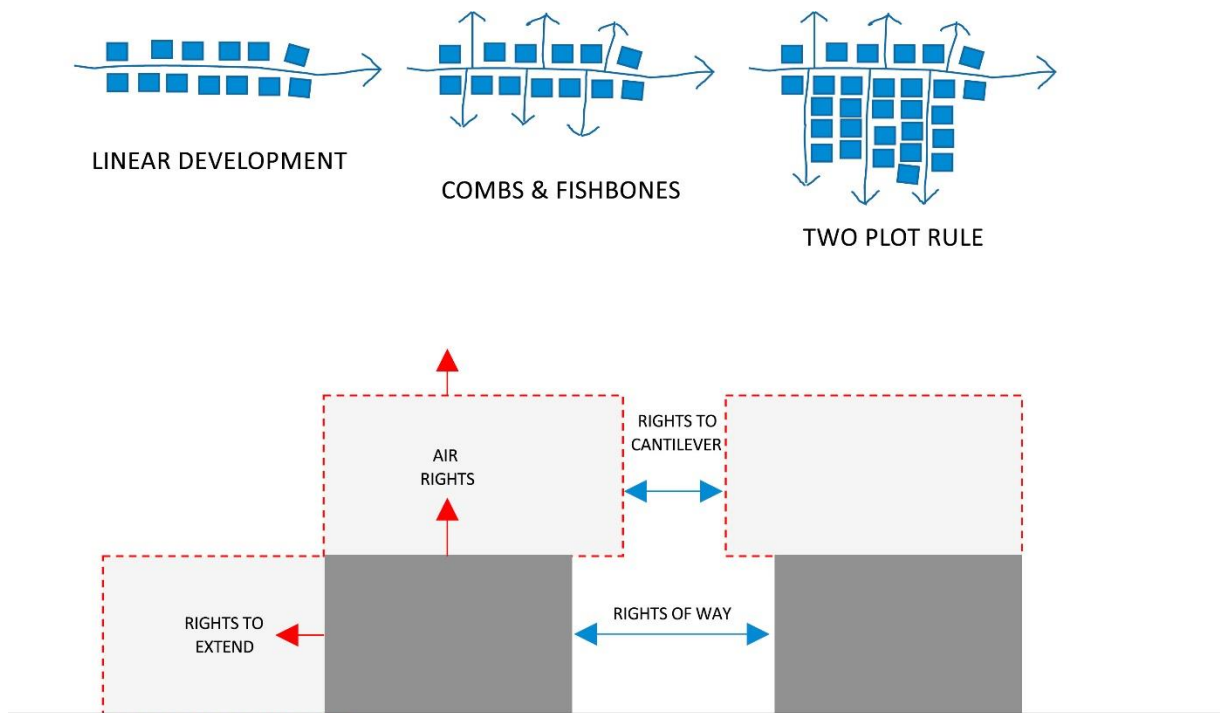


Figure 2.2 Informal building rights. Diagram by Kim Dovey. (Reprinted from Dovey et al., 2020, p.13)

informal modes of urban space production has the potential of delivering adaptive and economically sustainable urban fabric, therefore these two modes should not be viewed in opposition but rather in a balanced collaboration (Dovey, 2019). The study conducted on densely populated informal settlements of Bandung, Indonesia by Jones (2019) confirms the importance of the relations between formal order and informality in shaping the physical form of informal settlements. Jones (2019) uncovered a system of aggregation and adaptation that gives the complex urban fabric of these settlements, commonly viewed as chaotic and as a source of social problems, an informal self-regulated type of order.

As in the case of traditional settlements, informal settlements follow certain codes and self-organizing patterns:

- The informal morphogenic process is **incremental and co-evolves** as buildings and street network segments emerge as needed.
- The **right-of-way** is an implicit rule in any human settlement, ensuring continuous access to existing buildings. However, in the case of informal settlements, rights of way frequently become blocked through encroachment, allowing other routes to take over traffic (Dovey, 2019).
- **Encroachments**, usually taking the form of a single room or interface building elements such as porches or stairs (Dovey, 2019), take place as needed but within the limits of the right-of-way.
- **Views and air rights** are protected within the morphogenic process of informal settlements, and residents have air rights to the space above the last room they built, representing an opportunity for further encroachment and extra space (Figure 2.2).
- **Street segments with higher centrality** create a special interaction between private and public space within the settlement. **Street segments with lower centrality** could be encroached upon by buildings or even blocked, but walkable access to all structures is maintained.

- **Direct contact with a street segment**, guaranteed by the right-of-way principle, allows public space to be used as an extension of the private realm, creating opportunities for socialization and domestic production. The interaction between the private and public domains can be tailored to suit different domestic or public uses of space.

According to Jacobs (1961), These adaptive abilities are characteristic of informal morphology and the qualities they offer are typically lost within urban renewal projects aiming at formalizing these settlements.

### **2.1.3 Self-organized settlements as complex adaptive systems**

The study of self-organized settlements, be it traditional settlements or current informal settlements, yields the conclusion that their morphogenesis is characterized by aggregation of interventions from multiple actors in an ever-changing environment directed by adaptation to needs and nature-imposed constraints (Batty, 1971; Allen, 1997). An effective theoretical framework to conceive of self-organized settlement morphogenesis would be the concept of complex adaptive systems (Allen and Sanglier, 1981; Cecchini et al., 2000; Dovey, 2012). The concept was pioneered by John Holland and developed in his book titled “Hidden Order: How Adaptation Builds Complexity” (Holland, 1995). Complex adaptive systems (CAS) are defined as systems that involve multiple actors that adapt or learn as they interact between each other and the environment surrounding them. This framework sits at the heart of many contemporary problems related to simulating the dynamics within the natural world, including providing for sustainable human growth (Holland, 2006). CAS are systems where different elements are interconnected and interdependent on each other. They are systems with feedback loops that can lead to self-reinforcing or self-correcting behavior, leading to more resilience (Meadows, 2008). City growth is one of the most perceptible CAS instances. Buildings, structures, inhabitants, economic and cultural interactions are in constant change, and the urban morphology reflects that constant adaptation (Batty, 1971; Allen and Sanglier, 1981). As Holland (1995) put it: “A city is a pattern in time”. No actor within the urban environment remains the same, nevertheless, the human settlement continues to function and deliver the inhabitant’s needs through adaptation of

its morphology. The city is a system of organized complexity that exhibits dynamic and evolutionary behavior with an implicit morphogenesis (Jacobs, 1961). This fast-paced adaptation occurs without the constant presence of central planning thanks to the CAS self-organized way the city evolves while maintaining its coherence. CAS remains consistent in the face of change. They all constantly keep a certain organization and order. These systems are organized by a set of general principles that govern the interaction between the agents within it (Holland, 1995).

Within CAS, the various agents need to have freedom of action so that self-regulation and adaptation takes place. Within the urban realm, each inhabitant or household or organization has a set of possibilities and actions to choose from given the constraints imposed by the rights of the other actors within the environment, and the natural constraints present within the environment. Freedom of action is present, however, within a limited domain defined by the presence of other actors within the

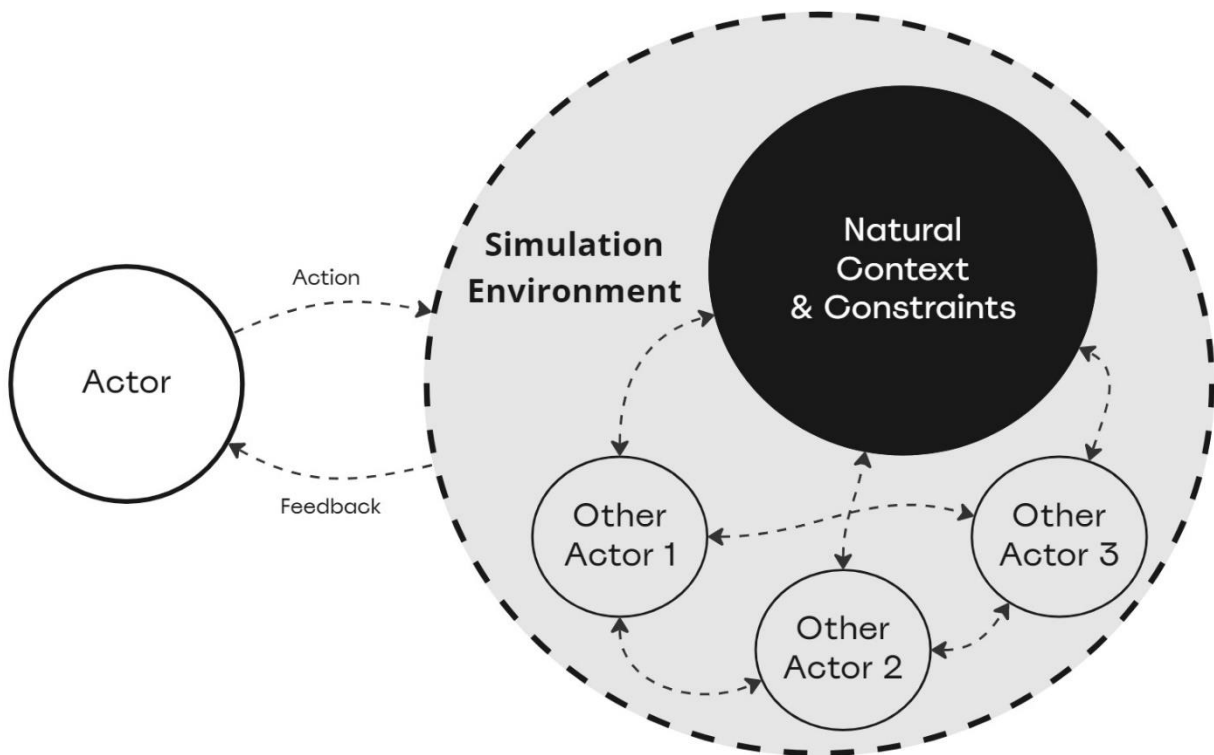


*Figure 2.3 CAS scenario visualized from the point of view of a single actor within a 4-actor simulation environment. The actor performs actions within the environment based on feedback received from the environment representing the effects of actions performed by other actors as well as natural constraints within that same environment. Produced by author.*

system who also enjoy a certain freedom of action. This characteristic of CAS gives birth to regulation of the set of actions and therefore the behavior of each agent within the environment (Holland, 1995). Thus, freedom of action insures adaptation and the regulation of the whole system.

The behavior of the agents within the environment is determined by a set of possible actions and a set of rules, but also by stimulus and reward. Each agent given their current position within the system in terms of possible actions and the current state of their neighboring environment seek to improve their condition. Therefore, they perform the sequences of actions that are expected to yield the most favorable outcome for them (Figure 2.3). Thus, the behavior within CAS is directed by a set of stimulus-response rules (Holland, 2006). Stimulus-response rules are conditional policies that the agent would take given the presence of a certain condition within the environment. In their most straightforward form, the agent constantly takes action A if condition C is present. However, CAS are more dynamic than that given the ever-changing nature of the environment within them. The same action would not constantly yield a positive outcome due to the intervention of the other actors within the environment. Therefore, the agents also keep changing their action policies given the same stimulus as they acquire experience (Holland, 1995). The agents therefore co-evolve through time and accumulated experience in a self-regulated, self-organized environment. Thus, when the range of possible observations or stimuli an agent can receive and the set of permissible actions, they can take at any point within the environment are clearly defined, it is possible to predict the behavior of that agent within that environment, and it is also possible to improve that behavior towards some predefined objective (Holland, 1995). Batty (1971) emphasizes the importance of incorporating feedback loops into urban simulation models, as cities are constantly changing in response to many factors. This stimulus-reaction loop within CAS (Figure 2.3) is compatible with the RL framework (Figure 3.3), as it is also directed by a feedback reward signal. This point is detailed further within chapter 3 and is central to the use of RL algorithms in modelling natural phenomena that have CAS characteristics such as self-organized settlement morphogenesis. Given this stimuli-response behavior within CAS modelling efforts must define the observation and reward signal carefully and define an action space that

allows for appropriate behavior by the agent (Holland, 1995). The sequence of actions and set of rules are not predefined by the modeller, but rather learned by the agent following a certain performance objective sought after by the modeller. Different stimulus-response combinations emphasize different aspects of the CAS, resulting in various models. As such, different definitions of settlement morphogenesis environments yield different models and different types and instances of settlement forms. It is not so much a question of correct or incorrect models as it is a question of what aspects and performance metrics are being investigated and optimized for (Holland, 1995).

The stimulus-response behavioral mechanism within CAS is a phenomenon that occurred within the morphogenic process of traditional towns. The actions of different actors contributing to these towns' morphogenesis were motivated and controlled by feedback (Hakim, 2014). This feedback is either positive or negative. Negative feedback regulates unpredictable disruptive interventions within the settlement enforcing equilibrium and equitability within it. While positive feedback reinforces favorable interventions making sure they repeat over time (Hakim, 2014). An example of feedback within settlements would be a simple interaction between neighbors where one of them decides to build a window that overlooks the private space of the other neighbor (Figure 2.1). Such action receives immediate negative feedback when the neighbor requests the window to be sealed or removed. Or if the window preceded the construction of the neighboring house, then the new neighbor's actions are geared by feedback to lay out the house in such a way that overlooking does not occur and pleasant views from the preexisting window are preserved (Hakim, 2014). The interaction between the various actors within the self-organized settlement system creates bounds to the actions performed by those actors at any given time. Thus, emergent systems, such as those in traditional urbanism, rely on operating within predefined boundaries. The ability of the system to learn and evolve is determined by its inevitable adherence to these boundaries (Hakim, 2014). Interaction between inhabitants, household, and local organizations and groups create feedback loops enabling the whole participants within the system to learn from their mistakes resulting in the adaptation of the settlements form over time.

Self-organized settlement morphogenesis is also distinguished by its dynamic nature, which stems from the dynamic decision-making processes that generate it. Top-down planning policies and zoning codes, on the other hand, do not allow for much flexibility. These are static rules that lack the ability to adapt to unexpected conditions through intelligence and feedback. Most interventions to formalize well-established informal settlements fail to meet the needs of the inhabitants because they lack the dynamic qualities offered only by the self-regulating generative processes of the informal mode of urban production. Current types of coding and conventional zoning dictate land use and enforces rigid regulations for each use and impose adherence to very specific provisions related to building form and settlement form which are static and lack dynamism and therefore cannot produce adaptability in settlement morphogenesis (Hakim, 2014). Dynamic rules and generative processes offer therefore a clear advantage over their static and descriptive counterparts in terms of adaptability, learning ability, and capacity to produce urban fabric that fits better the conditions present within the context.

To summarize, self-organized settlements exhibit CAS generative traits, allowing them to tap into continuous adaptability and learning, making them a more sustainable type of settlement as long as they are well regularized to ensure that the generative system is not abused by any actor resulting in the deterioration of the settlement (Dovey, 2019). The generative decision processes that rule self-organized settlement morphogenesis adhere to the following principles:

- **Freedom of action**, enabling individuals to act on their own knowledge.
- **Self-regulation**, the existence of a multitude of actors within the system simultaneously creates constant equilibrium within it.
- The system relies on **feedback**. Actions are reinforced or regulated through positive or negative feedback.
- The rules and codes of morphogenesis are based on a **set of performance objectives** on various levels. The individual actor's level to gear its

behavior, and the general performance of the settlement to suit the needs of all its inhabitants and adapt to environmental conditions.

- The morphogenesis follows a **generative program**, an **adaptive bottom-up system based on the decision-making of various contributing actors**, as opposed to a descriptive program that dictates a set of predefined, pre-engineered static rules and codes that cannot provide for any adaptability within the urban form.

The nature of the morphogenesis of self-organized settlements implies that the emergent forms and configurations would always be unique. The resulting morphologies would be adapted to each specific context. As a result, each locality of the self-organized built environment gains a stronger sense of place and identity (Hakim, 2014). These unique, adapted, and sometimes surprising outcomes will be visible from the level of an individual house design to the way clusters of buildings relate to one another establishing a certain character to the public space, access routes, and the entire settlement.

## **2.2 Digital modelling of self-organized settlement morphogenesis**

The first part of this chapter explored the characteristics and the nature of self-organized settlement morphogenesis based on theoretical morphological studies on traditional and current informal settlements. The morphogenic processes that produce urban fabric within these settlements are similar to the algorithmic codes that rule the growth within CAS phenomena. Many research efforts therefore are based on CAS simulation and optimization methods such as genetic algorithms to model and control the morphogenesis of such settlements. Within the second and current part of this chapter, the history of digital modelling within urban design is explored, as well as previous attempts of digital modelling and simulation of self-organized settlement morphogenesis using various methods.

### **2.2.1 The history of digital modelling methods in urban design**

Urban design is a discipline that tackles problems of incredible complexity due to the layered, numerous, and interconnected factors that influence the creation of urban form. The discipline had evolved throughout time, borrowing methods from other fields and adapting them to the urban design context to be able to handle and control the sheer complexity of urban planning and design problems. Since the advent of digital tools and processes in the 1950s, the formulation of the urban design problem and the process of planning of the creation fabric kept changing and improving according to the capabilities enabled by various computational and mathematical methods of the time.

The deployment of digital tools on urban design problems was first enabled by the capacity to accurately handle geometric shapes of various complexities, and aggregate and organize them in groups and layers. This possibility was facilitated by the increase of computing power and a wider availability of hardware during the 1980s. During that time, Computer-Aided Design (CAD) was born as a field of study. One result of the development of CAD programs and their integration into the design process was their impact on design thinking. The organization of CAD software led the design process into a new rationale of organization of design ideation into a systemic combination of basic modular components. The design process became directed by a series of steps to create and combine items to constitute the final proposal. The organization of the storage of the digital representation of the design into computer memory affected the conceptualization of the design itself to make it more compatible with the algorithmic nature of the tool of production used. During this same period when CAD programs were being developed and increasingly adopted within the design practice due to the increased availability of computer hardware, researchers were also developing theoretical and design experimental frameworks that are geared towards taking full advantage of the algorithmic problem-solving capacities that the emerging digital technologies of the time were offering. In the urban design domain, new frameworks to represent the spatial and social dynamics within a city and their impact on urban morphology started to emerge. New performance metrics such as network centrality started to be adapted from graph theory to represent urban morphological

dynamics, taking advantage of the digital representation of urban form. Space Syntax theory, first developed by Hillier and Hanson in 1984, emerged as a prominent method to model spatial properties as a result of social activities (Hillier & Hanson, 1984). Street networks started being represented as directed graphs, with nodes representing various important destinations within the urban fabric, and street segments represented as edges within the graph. This form and various other forms of digital representation of urban design problems enabled planners to leverage algorithms sourced from control theory to both expand their capacity for design space exploration, and to optimize urban form in order to maximize or minimize various performance metrics.

To perform such optimization on urban digital models, architects and planners needed more than just a digital geometrical representation offered by CAD software. Even though CAD permitted great control over complex geometries and helped planners with various repetitive drafting tasks, its limitations in terms of its ability to specify complex design rules and dynamics were obvious. Parametricism emerged as a new paradigm to fulfil these limitations and work in a complementary fashion with CAD technology (Schumacher, 2011), offering the ability to accurately represent geometry digitally as well as the ability to modify and execute algorithms of high complexity on these digital replicas of urban form. Parametric modeling modified the representation of any design prototype into a set of explicit rules or functions regulated by a set of variables or parameters. This representation gives designers the ability to algorithmically describe a prototype and all its possible variants. This form of representation permits the application of control theory methods by automatically changing the control parameters in the architectural or urban mathematical model definition to find the variant or set of variants that maximize or minimize a predefined performance objective. Parametric modelling methods have therefore addressed the key limitations of CAD techniques in terms of design problem representation. They gave designers the ability to rationalize design forms into strict sequences of rules that allow for ever-faster design space exploration and control due to the ever-increasing computational power available (Kolarevic & Klinger, 2008). The new capacities enabled by Parametricism in terms of design representation, modeling and thinking could well explain the widespread adoption of its methods within today's design practice. Designers could conceive of their

design problems more systematically and programmatically in compatibility with their capable computing tools. This shift gave rise to algorithmic design thinking where designers focus more of their time analyzing and formulating the underlying rules of a design to better fit the actual dynamics within the real world (Çalışkan, 2017), thus, enabling better design decision-making at scale.

However, parametrization of architectural and urban design problems appears to have peaked and saturated during the past decade and a few limitations of its use started to emerge. Most important of these limitations stems from the inability of parametric modelling methods to address design objectives that are not easily and readily representable in strict mathematical form. Moreover, the use of parametric modelling often falls short of delivering on its promise of efficient design space exploration due to the sheer size and complexity of most architectural and urban design problems. The use of such methods within a limited algorithmic representation of the problem renders the solution intractable and computationally expensive, which forces designers to resort to some oversimplifications within the model and therefore usually fail to form a fair representation of the reality of the design problem within the real world, and thus, fail to propose realistic solutions to design problems based solely on the parametric design approach. Another limitation of parametric design methods is their rigidity of representation, which fails to account for the dynamic nature of most real-world design problems, such as in the case of self-organized settlements for instance. Most real-world design problems follow adaptable rules, whereas parametric methods formulate and bound design problems in rigid predefined functions and algorithms.

In an attempt to overcome the limitations of traditional parametric design methods, research work to take advantage of the possibilities AI offers to architects, designers and planners have been intensifying in recent years. Quan et al. (2020) developed a design framework for smart urban design decision-making enabled by AI-aided Design (AIAD). This framework uses advanced AI search and control techniques to provide dynamic and AI assisted design decision-making procedures for urban design problems. Convolutional neural networks (CNNs) and Generative Adversarial Networks (GANs), types of deep neural networks (DNNs), were also heavily leveraged within

numerous research applications in architectural and urban design problems (Du et al., 2020; Huang and Zheng, 2018; Kelly et al., 2018; Nauata et al., 2020; Uzun et al., 2020; Wang et al. 2020; Zhang and Blasetti, 2020; Zheng and Yuan, 2021). These applications range from building floor plan generation, building 3D model generation, urban planning, and urban form generation, as well as leveraging of data available through Geographic Information Systems (GIS) and remote sensing data such as satellite imagery to predict the performance of the urban fabric especially in relation to environmental performance factors such as carbon emissions and energy efficiency. These applications all rely on an AI enabled machine-human complementarity in the design process, where AI assistants analyze the context and the data and make rough predictions that would then enable the human designer to reliably make better design decisions even within complex, dynamic, and non-linear design problems, and environments.

### **2.2.2 Attempts of self-organized settlement morphogenesis digital modelling**

Self-organized settlement understanding, and simulation work also benefited from and followed closely the rationale of development of digital modeling methods in urban design. Attempts at self-organized settlement morphogenesis simulation span the second half of the last century and extend all the way to the current decade with varying methods of representation and algorithmic control.

The first notable work in this regard is the work by Hillier and Hanson (1984). In their book entitled “The Social Logic of Space”, the two authors propose a model based on a simple cellular automata (CA) mechanism to represent the dynamics of settlement generation and growth. This model was labeled as ‘Beady Ring’ due to the type of settlement morphology it generated. The form of settlement named beady ring is notably found within small settlements of the Vaucluse department in France, where a sequence of small and large urban spaces were arranged around an irregular ring street, which gave the settlement the appearance of a set of beads on a string considering an overall plan view. This type of urban morphology was the result of self-organized modes of settlement generation where small-scale decisions were taken over time to result in that specific urban space configuration (Hillier and Hanson, 1984). The beady ring model is

a cellular automata aggregative model where urban form is represented by the different states a cell could be in within a two-dimensional array of cells. Three different states were allowed for each cell, either (i) a house cell, (ii) an open space cell, or (iii) a background carrier cell. This model is suitable for self-organized settlement morphogenesis due to its aggregative characteristic. Changes in cell states only happen incrementally at the cells located on the edges of the growing settlement representation. The logic of this model is used by the current thesis as a base reference for some of the RL environments developed for the experiments set forth in chapter 4.

The work presented by Erickson and Lloyd-Jones (1997) built upon the beady ring model and previous works within urban morphology theory and settlement digital modeling during the 1980s and 1990s to produce a model named 'Path Node Generator' that is more flexible and more accurately represents the dynamics of settlement

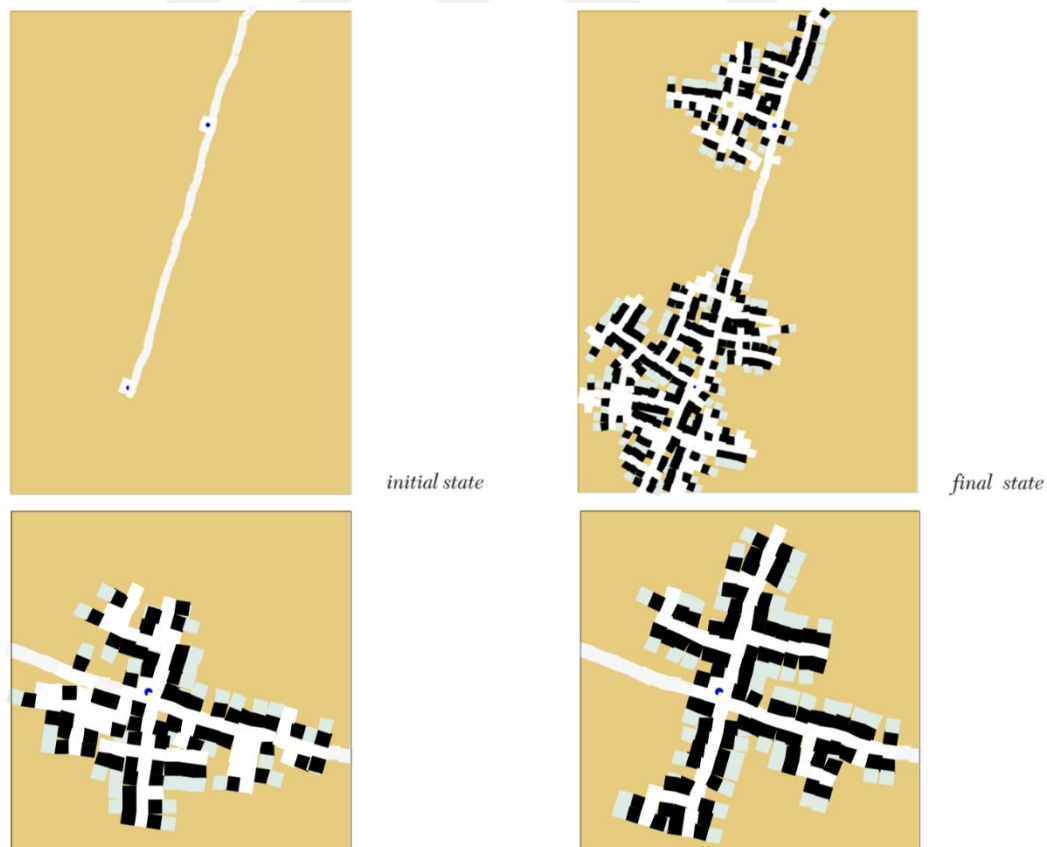


Figure 2.4 Sample output of settlement simulation generated by the 'Path Node Generator' model. (Reprinted from Erickson and Lloyd-Jones, 1997, p.919)

morphogenesis at the local scale of unit of aggregation. This model consists of cells representing five types of space, a carrier space (orange), two categories of urban circulation space (white), street cells and road cells, as well as two types of private space, house cells (black) and garden cells (grey) (Figure 2.4). Street cells are the most important cells within this model. They direct the aggregation process since private house cells can only be added in a cell adjacent to a street cell to maintain access rights within the settlement. The house cells also follow the orientation of the street cell they are attached to. Road cells act similarly to street cells except they do not offer an opportunity for aggregation but rather represent the initial condition of the site of preexisting infrastructure. Garden cells are always attached to the rear of each house cell to form the basic building type that forms the fundamental aggregation unit of the settlements generated by this model. The aggregation process is done incrementally at a randomly selected cell. If the chosen cell is a carrier cell, then a change to the state of that cell is performed. Both the orientation of the cells and the nature of the cell introduced at each step are dictated at random by preset probability distributions. A random noise between  $-5^\circ$  and  $5^\circ$  is also added at random to the chosen cell orientation at each step to introduce some irregularity to the street network and urban massing configurations generated by the model (Figure 2.4). The setup of these probability distributions was intuitively chosen by Erickson and Lloyd-Jones (1997) to result in urban morphologies that closely reflect the irregular and loose orthogonal circulation patterns found in many real-world informal settlements. These probability distributions have great control over the outcome of the morphogenesis simulation, and therefore, should not be rigid to better depict the dynamic changes that occur within real self-organized settlements. These probabilities of taking a certain action over another given a certain context should be drawn from feedback information coming from the surrounding environment of that local action within the settlement. These action policies should be learned and optimized to be able to be adaptable to change given certain preset objectives for the outcome wanted for the settlement's morphology and its emanating performance. Erickson and Lloyd-Jones (1997) noted within their work that the 'Path Node Generator' model could be improved by including feedback measures such as street network connectivity and relative asymmetry. This type of feedback could be used

to adapt the rules of aggregation of the model (Erickson & Lloyd-Jones, 1997). This thesis investigates this possibility by adapting this model into RL environments and deploying RL agents to learn a certain policy of action, a probability distribution over all possible actions at each step, to generate settlements and simulate self-regulated urban morphogenesis given preset settlement performance objectives formulated as a feedback or reward signal at each step.

The availability of algorithmic rule-based techniques in the generation of complex geometric forms developed during the second half of the 20th century continued to be exploited within self-organized settlement morphogenesis modeling during the first two decades of the current century. These techniques, such as cellular automata, aggregation models, and shape grammars, all function through the repetitive execution of a group of preset simple algorithmic transformations (Erickson & Lloyd-Jones, 1997). The understanding developed through the analysis of real-world traditional and informal settlements was used in combination with the available computational power that 21st century hardware offered to simulate the morphogenic behavior of these settlements (Dovey et al., 2020). Many notable attempts at modeling informal settlements' morphogenesis can be found within recent literature. Some of these models used cellular automata and agent-based modeling techniques that gained in popularity during the 1990s (Heath et al., 2009; Heppenstall et al., 2016, Hillier et al. 2000). Others used shape grammars for a rule-based aggregation of settlements using combinations between a vocabulary of unit components of these settlements (Barros et al., 2013; Dias, 2014; Duarte et al., 2007; Rahimian et al., 2020; Verniz & Duarte, 2017).

Duarte et al. (2007) conducted research aimed at creating a parametric urban shape grammar for the Moroccan Medina of Marrakech. The goal was to lay the groundwork for a system that could capture aspects of the existing urban fabric and apply them to other cases of urban morphogenesis simulation to help with future urban interventions. The methodology employed started with historical analysis and fieldwork to then progress to the identification of three sub-grammars required to encode the complexity of urban generation processes within that specific traditional settlement: (i) the urban grammar, (ii) the negotiation grammar, and (iii) the patio-house grammar.

These three processes handle three different levels of scale of intervention, first one (i) dedicated to the arrangement of access routes and urban blocks, the second one (ii) plays the role of a transition between both other grammars to regulate and adapt changes occurring at both levels of urban intervention, and the third and last grammar (iii) handled the interior functional organization of houses within the settlement. Subsequent research also employed this shape grammar method described to analyze different informal settlements around the world and extract their generative logic. For instance, Barros et al. (2013) used a similar shape grammar method to analyze the evolution of Maputo's informal settlements. This study focused on understanding how encroachments of single room units manage to create an informal settlement on a preexisting formal street network. The focus in this study was on a self-organized settlement instance where urban planning interventions precede architectural massing concerns. Building generation adapts to an established street network pattern. Dias (2014) also used a shape grammar method inspired by the work of Duarte et al. (2007) to encode the morphogenic process of informal settlements in Rio de Janeiro, Brazil to enable future interventions within those favelas with better understanding over the dynamic interactions that result in those settlements' forms. The settlement generation process was simulated using 13 rules for street network generation and 10 other rules to control building composition. The repetitive combination of these rules resulted in many variations of settlements that resemble the morphology of the Rocinha favela informal settlement. More recently, Verniz and Duarte (2017) explored how spontaneous occupation in informal settlements evolves, and how to plan new settlements considering topographic conditions. This paper focused on the use of a parametric urban shape grammar as a methodology to describe the complex urban form of informal settlements, explaining how buildings and pathways create a maze-like urban structure, and how different building typologies are located according to many factors like topography, the urban context, and the functional organization of these buildings. This research describes an urban shape grammar method for the simulation of informal settlement morphogenesis in steep terrain areas, which is the most common case of topographic context when it comes to informal settlements that few studies within the literature consider.

With improvements in computing power and speed and increases in data storage capacity, researchers within the urban digital modeling domain started to shift their efforts towards agent-based modeling (ABM) techniques. ABM methods enable more autonomy and freedom of action of individual agents within urban modeling environments, which at the micro-level of simulation more accurately fits the type of dynamics found within real-world urban morphogenesis scenarios (Heppenstall et al., 2016). Many researchers started to adapt CA models to enable more freedom of action at the basic individual scale of intervention within urban scenarios. Bura et al. (1996) is one of the first urban simulation works to identify the potential of ABM methods in urban growth modeling. This work recognized that the simulation of the evolution of any system of settlement growth and its dynamics must consider all of the components

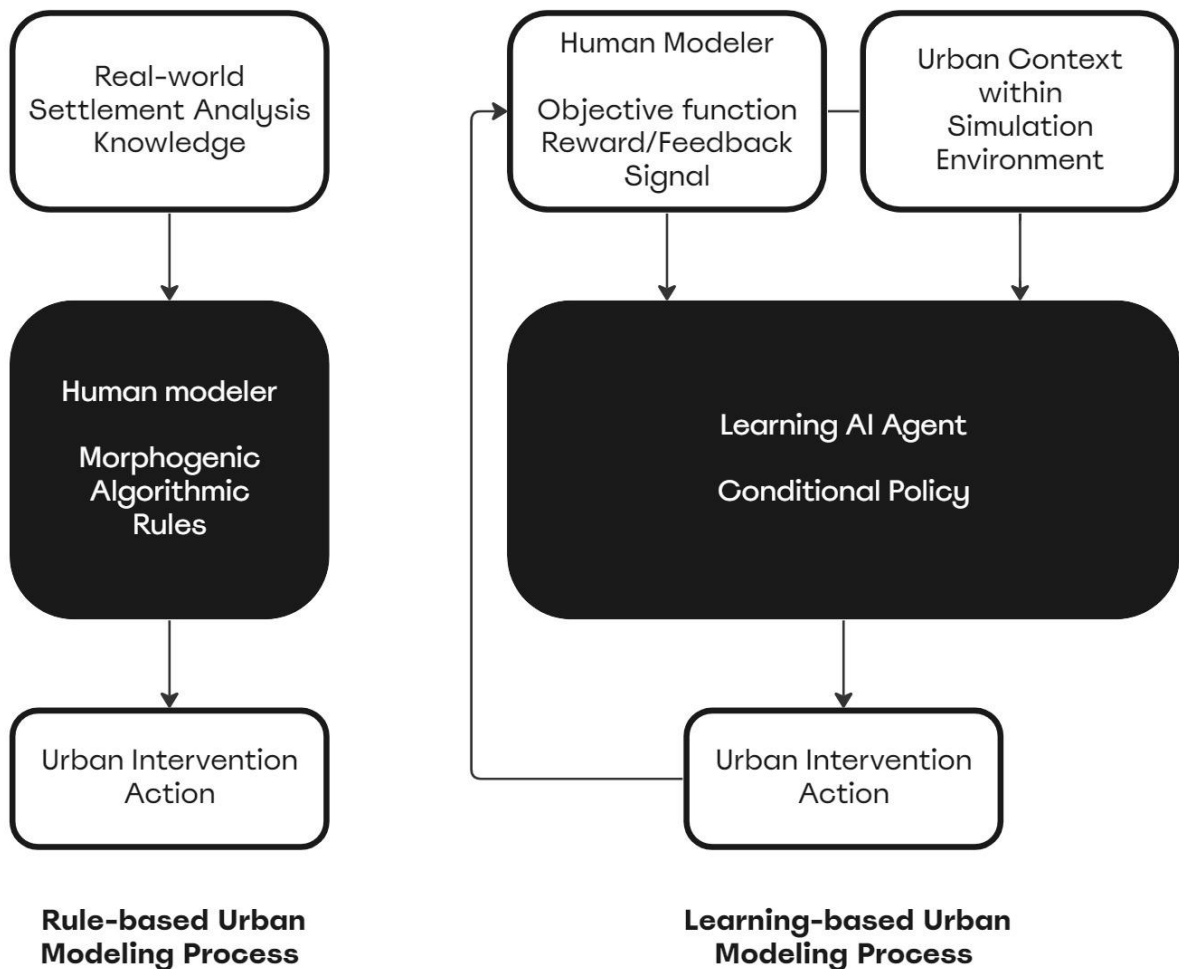


Figure 2.5 Rule-based urban settlement modeling approach as opposed to the proposed learning-based urban modeling approach. Produced by author.

in action resulting in spatial aggregation of settlements and the complex interactions of cooperation and competition within urban activities continuously occurring within settlements. This work found multi-agent simulation (MAS) methods to be fitting for this simulation task. Batty (2005) reinforced this orientation in viewing urban growth and its dynamics and the way it should be approached by modeling and simulation efforts.

ABM methods offer an improvement over models that take a macro scale general view to urban growth dynamics (Koenig et al., 2020; Koenig & Varoudis, 2016; Liu, 2020; Miao et al., 2020; Qeisi & Al-Alwan, 2021). The morphogenesis of settlements is a direct result of the adaptation between all the interventions occurring within the same environment. Therefore, a more fitting simulation must represent all these actions taking place at the micro scale individual level, and at the most fundamental level of the unit of urban spatial aggregation.

### **2.2.3 Limitations of rule-based methods and the potential of learning-based methods in settlement morphogenesis simulation**

The current research in modelling and digital simulation has promise and obvious benefits, but such techniques, paradoxically, rely on a deeper domain knowledge of settlement morphogenesis dynamics to be able to design the algorithm followed to simulate such settlements (Dovey et al., 2020). Therefore, these works are only digital representations of knowledge about algorithmic rules of growth of specific self-organized settlement cases that researchers developed through careful observation and analysis of real-world settlements. The rules within them are imposed by human experts and are not autonomously learned. Therefore, they are not adaptable to new settlement scenarios, and are limited to simulating the morphogenetic process of a single type of settlement and their algorithmic rules cannot be generalized for any other similar context or condition. It is however important when engaging with complex systems to have an adaptive, flexible, and conditional set of rules and policies (Meadows, 2001). A policy that is constantly learning and adjusting to the changes occurring in the environment. Using the previously mentioned methods for urban growth simulation, the modeller needs a priori knowledge of the specific algorithmic rules that would govern

the actions of each component within the simulation system (Figure 2.5). It is, however, a rather complex task to uncover these rules for the specific settlement in question of its specific context each time an attempt of settlement morphogenesis simulation is under way.

To overcome this limitation and to be able to use computational methods to help uncover self-organized settlements' morphogenic rules and codes, these models must be learning models, able to develop meaningful knowledge and representations of a settlement's generative dynamics, and therefore, able to simulate such morphogenesis and capable of generalizing to new settings and contexts by employing the learnt conditional generative policies (Figure 2.5). This thesis proposes to formulate self-organized settlement generation as a Markov decision process (MDP), as will be explained in the following chapter, and apply DRL methods to learn the morphogenic policies of self-organized settlement generation through interaction with a simulated environment with a set of predefined objectives and performance metrics. The learning agent in this case is encouraged to develop urban intervention policies that avoid unfavorable urban conditions while reinforcing desirable ones based on the feedback it receives. This feedback signal must be carefully designed by the modeler or the planner to serve the objective intended for the generated settlement's performance. Designing this reward or feedback signal is a much easier task for a human designer compared to the deduction of the algorithmic logics that govern settlement morphogenesis in each simulation case.

## **CHAPTER 3**

### **3 Reinforcement Learning in Design**

The previous chapters have reviewed research related to self-organized settlement morphology and its simulation using various computational methods. The theoretical reasoning behind the pertinence of the use of learning agent-based methods, as opposed to rule-based methods for the modeling of self-organized settlement morphogenesis was also established. These methods, within their theoretical definition and modes of operation, are more adequate and more coherent with the nature of the morphogenic process of urban growth. Reinforcement learning (RL), and more specifically deep reinforcement learning (DRL), is a prominent learning agent-based simulation framework. This chapter will detail the fundamental components of this framework and explain how it could be used for self-organized settlement morphogenesis modeling.

#### **3.1 Reinforcement learning fundamentals**

This section will situate RL within the broader field of artificial intelligence (AI) and explain how it relates to other subdomains within this field. It will also serve as an introduction to some of the important fundamental components and concepts within RL. Therefore, laying the theoretical groundwork for the introduction of a DRL modeling framework for settlement morphogenesis in later sections of the current chapter. This framework would then be employed within practical experiments of self-organized

settlement morphogenesis simulation in chapter 4 of this thesis.

### **3.1.1 Artificial intelligence, machine learning, and deep learning**

Artificial intelligence (AI) is a field of study that is concerned with the replication of human-like intelligent attributes and capabilities, such as reasoning, decision-making, and the ability to understand natural language, within computational machines. The field of AI owes much of its beginnings to Alan Turing. Within his seminal paper titled “Computing Machinery and Intelligence”, Turing (1950) defined what is known now as the ‘Turing test’ or ‘the imitation game’, which is a test that aims at identifying whether a digital machine, that is able to perform any computational operation, is displaying reasoning or behavior which could be described as intelligent, as in, behavior that could not be differentiated from human behavior and reasoning, thus, the machine in that case would have intelligence that is equivalent to human intelligence.

Machine learning (ML) is a subdomain of AI (Figure 3.1). ML is the subfield that is responsible for most successes of AI within research and industry and is the subfield that receives most attention and concentration of efforts within the AI research community. ML is a field of study that aims at processing empirical data using various computational algorithms to create usable mathematical models to make predictions, detect anomalies and identify patterns within a dataset, cluster data according to their distribution, classify data into preset categories, or assist with decision-making using statistical principles. ML enables machines to learn mathematical function parameters, also referred to as weights, using input training data and following the gradient direction that minimizes an objective function referred to within the ML community as the loss function. This process of learning gives machines the capacity to learn representations or algorithmic rules without the need of explicit programming and implementation by a human programmer.

ML consists of three fundamental frameworks (Figure 3.2). First, supervised learning, which is the kind of learning featured in most current research in ML, deals with training models to make predictions using labeled examples provided by a knowledgeable external supervisor or expert (Sutton & Barto, 1998). Each example is an encoding of an input situation along with a representation of the correct action the model should take in that situation, also known as the label. The purpose of supervised learning is to extrapolate or generalize to new input situations not available and not seen before within the training set. This category of ML is not suitable for self-organized settlement growth modeling due to the difficulty and expensive computational cost of building a balanced labeled dataset. It is also not suitable for this task because it is actually impossible to provide expert labeled data in cases never seen before for the human supervisor. Self-organized settlement morphogenesis simulation is an interactive problem by nature; thus, it is impractical to obtain labeled examples of the desired

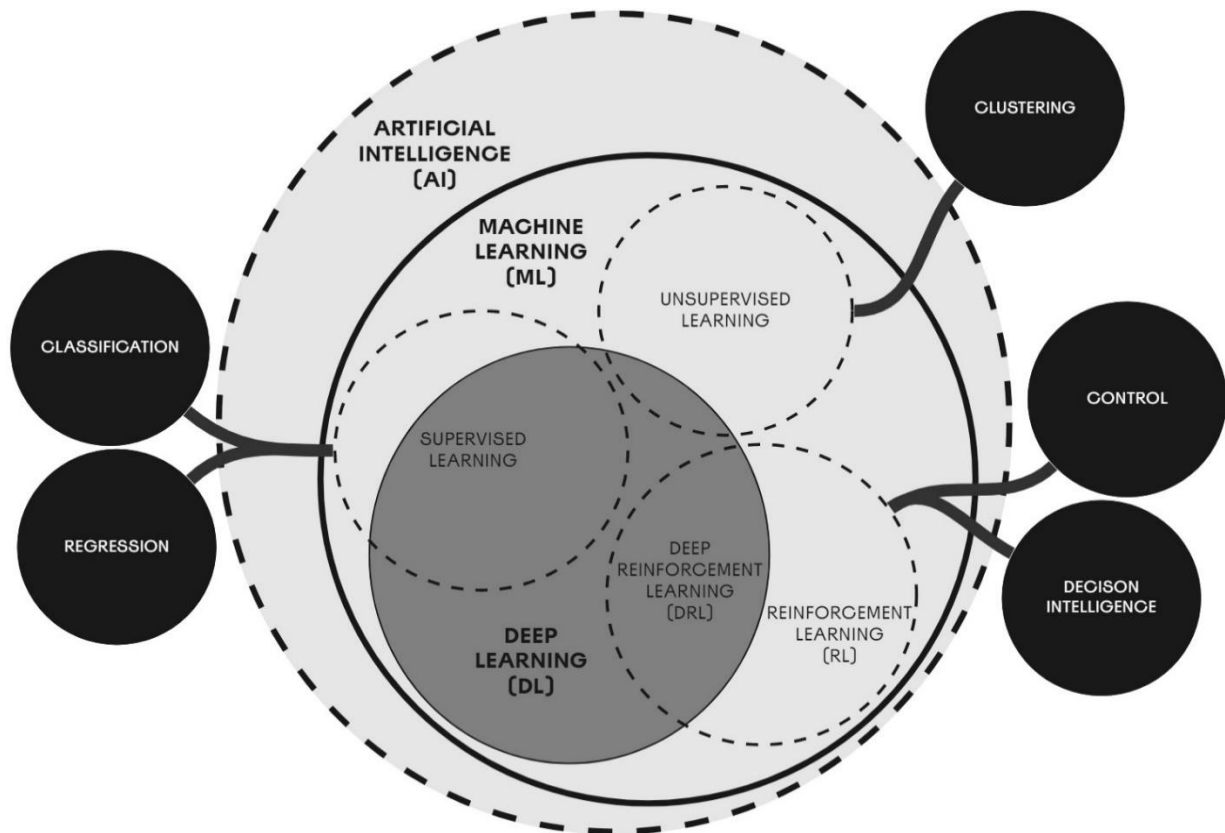


Figure 3.1 The relationship between artificial intelligence (AI), machine learning (ML), and deep learning (DL) and their different subdomains as well as the different categories of tasks they are most suitable for. Produced by author.

behavior for each step of the simulation. In this kind of interactive problem, the agent must be able to learn iteratively from its own experience (Sutton & Barto, 1998).

Unsupervised learning, on the other hand, trains clustering models on unlabeled data. It has no clear objective; the only aim of this framework is to uncover statistical relationships between various data points. Thus, this particular framework is not suitable for settlement morphogenesis simulation where the design problem involves various clear objectives of performance and adaptation to socioeconomic factors as well as natural environmental constraints.

The third and last framework within ML is reinforcement learning (RL). Within this framework an agent is trained to find an optimal policy within an environment. The agent learns this policy through the interaction with this environment by taking actions and consequently receiving a reward signal in addition to an observation of the environment's new state in a continuous iterative process (Silver, 2015). The objective of the agent within this ML paradigm is to maximize a reward signal. RL could be thought of as a form of unsupervised learning since it does not require correct ground truth labels; however, identifying structures and patterns within input data does not satisfy an RL agent's objective of maximizing reward (Sutton & Barto, 1998).

Uncovering patterns in the observation signal of an agent can surely help the agent find better policies that would yield higher reward, but it is not sufficient alone. RL also relies heavily on ideas and algorithms used within supervised learning. In fact, an RL agent's policy could be viewed as a supervised learning model that updates its parameters at each training step based on the gradient of a loss function that is negatively correlated with the reward function. However, it is technically not a supervised learning model because there is no involvement from any expert supervisor telling the agent what constitutes correct behavior. The agent rather learns from its own experience of interaction with its environment with the aim of maximizing a certain preset reward function. RL comprises of different families of algorithms such as Q-learning, policy gradient algorithms, as well as actor-critic methods. These different algorithm classes are classified into three categories, namely *policy-based*, *value-based*, and *actor-critic* classes of algorithms. This classification is done according to whether

these algorithms are geared towards learning a function that represents the probability distribution of taking a certain action given a certain state of the environment (policy function), or they are designed to learn a value function that determines how valuable a certain state is based on a prediction of future expected reward starting from that state (value function), or they are setup to learn both the value function and the policy in the case of actor-critic methods. Within the experiments conducted in this thesis, both policy-based and value-based methods are considered. First, the Deep Q-Network (DQN) algorithm introduced by Mnih et al. (2013), which is a value-based approach that learns an action-state value function represented as a neural network and referred to as the Q function to then choose the actions that have the highest value at each state to achieve an optimal policy of action within a certain environment. In addition to the Proximal Policy Optimization (PPO) algorithm introduced by Schulman et al. (2017), which is a policy-based approach that aims at learning the policy function directly represented as a neural network, by gradually updating the weights and biases of the neural network according to the rewards perceived from the environment after each action. The PPO algorithm is an improvement over vanilla policy optimization methods that preceded it since it adds a clipping mechanism to each gradient update which leads to lower variance during training and therefore a more stable and robust learning procedure.

RL has a clear conceptual alignment with generative design when viewed as a sequential design decision process that leads to different, sometimes unexpected, generative design proposals (Wang & Snooks, 2021). The sequential and iterative mode of training of RL agents without the need for supervision is compatible and suitable for modeling and controlling the constantly evolving generative design environments such as the self-organized settlement morphogenesis environments addressed in this thesis.

Under machine learning also falls deep learning (DL). It is a learning paradigm where the model is a deep neural network. The paradigm was first proposed by Ian Goodfellow, Yoshua Bengio, and Aaron Courville in their seminal book titled ‘Deep Learning’ (Goodfellow et al., 2016). The mathematical concepts behind this paradigm, especially neural networks, existed way before the publication of this book and could be traced all the way to the last couple decades of the 20th century, and even earlier. DL is a field of study that focuses on mathematical learning using artificial neural networks (ANNs) with many hidden layers, hence the name of the field. ANNs are systems of computational nodes, referred to as neurons, that perform linear computations with two parameters called weights and biases. These artificial neurons are hierarchically organized in consecutive layers that are linked together with non-linear operations. With this conceptual mathematical organization that loosely mimics the functioning of the human brain’s neural circuits, ANNs are able to learn complex non-linear mappings between input data and an output prediction. This, along with the increased of availability and performance of GPUs in the last decade, explains the enormous success of DL applications in solving many real-world learning problems, such as self-driving vehicles, gaming AI, robotics, and recommendation systems (Henderson et al., 2018).

Deep reinforcement learning is the type of RL where the policy function or the value function or both are ANNs. By leveraging the power of deep learning, DRL methods are capable of learning policies within complex, dynamic, and non-linear simulation environments such as the ones imposed by most generative design problems, and self-organized settlement generation is no exception.

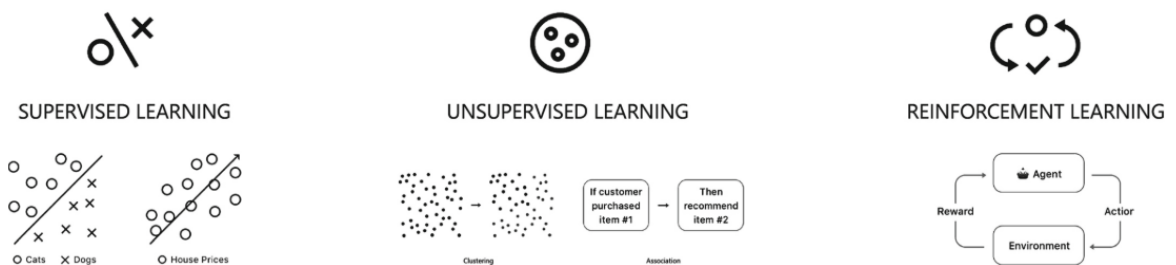


Figure 3.2 The three fundamental frameworks of machine learning. (Reprinted from Wang and Snooks, 2021, p.191)

### 3.1.2 The reinforcement learning framework

Learning by interacting with an environment is the most common form of learning among all intelligent beings on the planet. RL is a form of learning that incorporates this idea of interaction coupled with a clearly defined goal. All RL agents learn towards an explicit learning target, they can influence their environments through their actions and receive signals from that environment in the form of feedback and observation. Most of the ideas, methods, and algorithms of RL mimic, even though loosely, evolutionarily adaptable biological learning systems (Sutton & Barto, 1998). RL is the closest form of machine learning to the modes of learning operated by humans, animals, and other natural intelligent organisms.

RL aims to develop a strategy of action given the present situation the agent finds itself in, in order to maximize a numerical reward signal. The RL framework is a closed recursive loop where an agent performs actions that influence its environment, which in its turn changes in state and returns a response to the agent that then influences its next action and so on (Figure 3.3). Through this exercise, the agent accumulates experience and starts figuring out what actions lead to what responses from the environment, and therefore, gradually improves its policy of action relative to the responses it receives to maximize its reward over the long run. Challenging RL problems are the ones where following a certain policy of action that yields high immediate reward does not necessarily guarantee high accumulated reward over time. For instance, choosing to capture a minor piece in chess might look attractive considering immediate gains, but could, in some positions, give the adversary some advantage that could then result in a losing game in the long term. The challenge facing any RL agent then is to learn and capture the important aspects of the responses it receives from the environment through trial and error, in order to gain understanding of what actions are more favorable than others, in terms of achieving its goal, given any particular state its environment is in. There are therefore three crucial elements to the RL framework: (i) the information served to the agent about its environment, (ii) the action that agent takes given that information, and (iii) the goal that agent is trying to achieve formulated as maximization of a reward signal.

The RL framework is composed of many other main elements other than the agent and the environment (Figure 3.3). The RL system functions through a policy, a reward signal, as well as a value function. The policy determines the agent's behavior at any given time (Sutton & Barto, 1998). It is a function that maps the observation of the state of the environment to the actions that the agent could take. This mapping determines the conditional and adaptive behavior of the agent based on the response received from the environment. This formulation of the policy within RL is equivalent to the stimulus-response behavior noted within CAS and self-organized settlement morphogenesis discussed in chapter 2. The policy is the most important component of the RL framework because it could alone be sufficient to describe the agent's behavior (Sutton & Barto, 1998). It could take the form of a simple conditional look-up table, or it could even be a stochastic policy where the agent takes actions according to a learned conditional probability distribution over the set of all possible actions. The reward signal is also a crucial component within the RL framework. It determines the goal that the agent's learning would be directed towards. The reward signal defines what constitutes a bad or a good situation for the agent (Sutton & Barto, 1998). This could be thought of as an indication of good quality or bad quality of urban space in the case of self-organized settlement modeling. The reward directly or indirectly depends on the agent's chosen action at any timestep, as well as the state the environment transitioned to at that same timestep. The agent can only affect its reward through its actions at each step and does

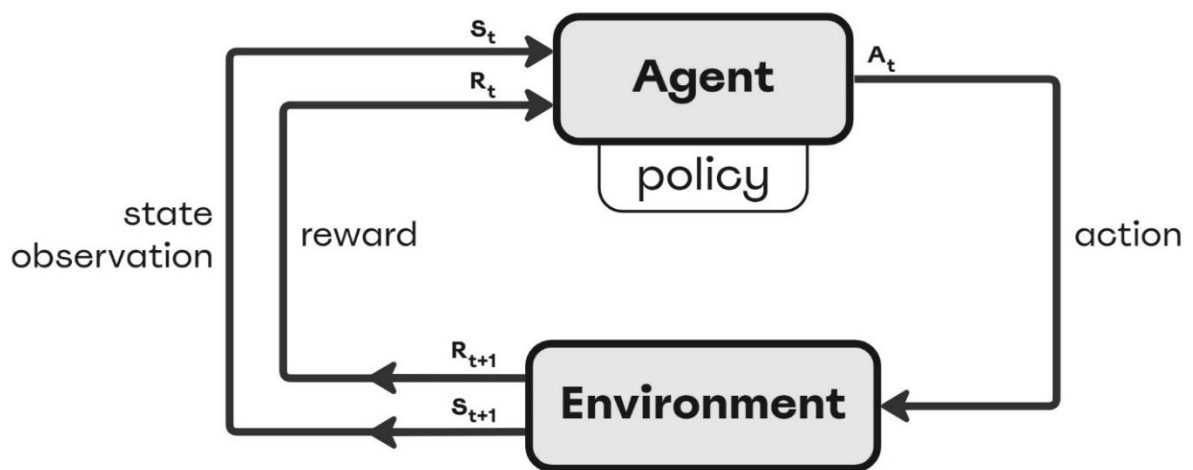


Figure 3.3 The agent-environment interaction in reinforcement learning. Produced by author based on Sutton & Barto, 2015.

not affect the logic of reward within the environment. That logic is set by the modeler and should accurately reflect what is positive or negative according to the goal of any modeling attempt. The policy of the agent is optimized and altered based on a maximization process of this reward signal (Silver, 2015).

An important challenge within reinforcement learning is the exploration-exploitation tradeoff. While learning, an agent must discover and try out new actions to uncover whether they are effective and yield higher rewards. However, that same agent must also exploit the knowledge it already learned and accumulated through experience about certain seemingly favorable actions to get high rewards. This imposes a balancing act on the agent while learning because it cannot maximize reward while pursuing any of these two strategies exclusively (Sutton & Barto, 1998). In general, the agent must explore a variety of options initially and gradually increase its exploitation rate as it gains more confidence about which actions are more favorable at each given state. This dilemma is called the exploration-exploitation tradeoff within decision theory, and various RL methods were developed particularly to tackle this challenge, one of the most basic and widely used of which is the  $\epsilon$ -greedy algorithm (Mnih et al., 2015), where the agent takes an exploratory action with probability  $\epsilon$  and a greedy action with probability  $1 - \epsilon$ . This exploration-exploitation dilemma is also at the core of design decision-making. An architect or urban planner could rely on previously tried techniques and solutions to provide a proposal to a current design problem and guarantee a certain level of success, but they should also try new methods and technologies to make sure they are constantly producing adequate design solutions of the highest quality and standards over the long run.

RL is an approach to artificial intelligence that is based on a few conceptual principles coupled with domain knowledge that is fed to the system through the setup of the reward function and the environment. The structure of the RL framework reduces the need for a large number of task specific procedures and heuristics to achieve meaningful behavior within a simulation (Sutton & Barto, 1998). Previous attempts at self-organized settlement morphogenesis digital modeling, and urban form modeling in general, such as the ones discussed within chapter 2, rely on such task specific exact procedures and

algorithmic generative rules which are difficult to formulate for every type of settlement growth dynamics. This thesis proposes the use of RL framework exactly to take advantage of its general learning characteristics. Following this framework, self-organized settlement morphogenesis modeling would stay constant in its operational components and only change in terms of the specific domain knowledge fed to it through the reward signal and the setup of the environment to define the objectives and units of settlement growth for each specific settlement case. The rules of sequential aggregation would be learned by the agent as a conditional policy to serve a set of predefined objectives.

### **3.1.3 Markov decision processes and reinforcement learning**

RL methods aim at learning how to optimally behave within an environment. Markov decision processes are a formal rigorous description of such environments. Within this section, a detailed description of what constitutes an MDP is laid out. The conceptual framework of MDPs is essential to this research and is used as a base to extract the set of components that should constitute any generative design environment and their characteristics, to be able to take advantage of DRL methods in generative simulation, design optimization, and design space exploration.

Within RL, the agent takes decisions based on information on the state of the environment it receives at each step. In order for RL theory and methods to apply and function on such environments, this observation or representation of the state of the environment at any time step should have what is called the Markov property (Sutton & Barto, 1998). For a detailed explanation and a formal mathematical definition of the Markov property refer to the appendix.

Any RL task that satisfies the Markov property is called a Markov decision process (MDP). An MDP with finite action and state spaces is called a *finite MDP*. Most modern RL methods are geared towards solving *finite MDPs* (Sutton & Barto, 1998). This thesis focuses on the modeling of self-organized settlement morphogenesis as a *finite MDP*. Any *finite MDP* is defined by the tuple  $(S, A, \mathcal{P}, \mathcal{R}, \gamma)$ , the definition of its state space  $S$ , action space  $A$ , an objective or immediate reward function  $\mathcal{R}$ , a discount factor  $\gamma \in [0,1]$  that is a parameter that represents the level of importance attributed to long-term future reward, and the one-step dynamics of the environment  $\mathcal{P}$ , meaning the probabilistic dynamics of transition between states (figure 3.4) (Sutton & Barto, 1998). More formally, the transition between states is governed by a probability function  $\delta$  that gives the probability of landing in a state  $s_{i+1}$  starting from a current state  $s_i \in S$  and taking an action  $a_i \in A$ , i.e.  $\delta(s_i, a_i)(s_{i+1}) = \Pr(s_{i+1}|s_i, a_i)$  (BuHamdan et al., 2022).

MDPs are a way to model decision-making within a discrete, complex, and dynamic sequential environment, such as the ones found within most urban design and

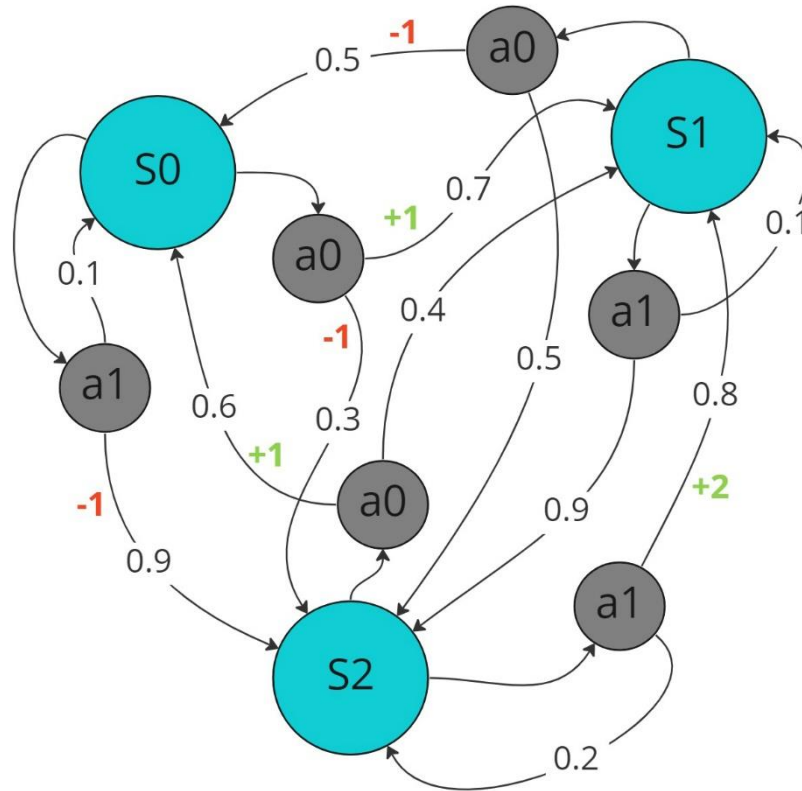


Figure 3.4 A simple MDP example with three possible states and two allowable actions, with transition probabilities from one state to another, and resulting reward signals given the current state and the action chosen by the agent. Produced by author.

planning problems including the modeling of settlement growth. A solution to an MDP is given as a policy  $\pi$  that provides an action to take in whatever state an agent might find itself in while maximizing the expected cumulative reward at each step. The policy that leads to the maximum possible cumulative reward is called an optimal policy  $\pi^*$ . Each MDP might have multiple optimal policies. To achieve or get close to an optimal policy, an agent must accumulate experience within the environment. This experience comes in the form of sequences of trials and errors that are called Markov chains or trajectories (Figure 3.4) i.e.  $s_0, a_0, r_0, s_1, a_1, r_1, \dots, s_{n-1}, a_{n-1}, r_{n-1}$  for a trajectory with a length of  $n$  steps.

### 3.2 The Markov decision process for generative design problems

Generative design is a design process that aims at exploiting the generative capabilities of computational methods to automate the generation of design proposals, enable an efficient exploration of the design space, and quickly find suboptimal to optimal solutions to design problems given a predefined design objective. This process involves the generation, evaluation of design alternatives according to performance metrics, and then the selection of the best performing options for a declared design problem. The generation of these design alternatives is done through the computational execution of predefined geometric, spatial, and parametric algorithmic rules and grammars, as reviewed in chapter 2, and the search and optimization part is handled using various heuristic methods, search algorithms, genetic algorithms, and machine learning methods more recently. Genetic algorithms are found to be less sample efficient and require more computation time in finding optimal solutions to optimization problems compared to deep reinforcement learning methods. The gap in efficiency between these two families of algorithms gets even bigger when the number of possible trajectories gets larger (Luis et al., 2021; Ororbia & Warn, 2022; Wibranek et al., 2021). Most real-world generative design problems involve extremely large trajectory spaces. This explains the surge in DRL adoption to tackle generative design problems within recent years, some of these efforts replace earlier proposed methods based on genetic algorithms which are predominant within generative design practice today (Makki et al., 2018; Rutten, 2013; Vierlinger, 2013).

From a decision-making point of view, generative design could be viewed as a sequence of decisions or changes and interventions on the design proposal that aim at improving it towards satisfying a certain design objective. This decision-making process could be modeled using MDPs, where the states would correspond to design

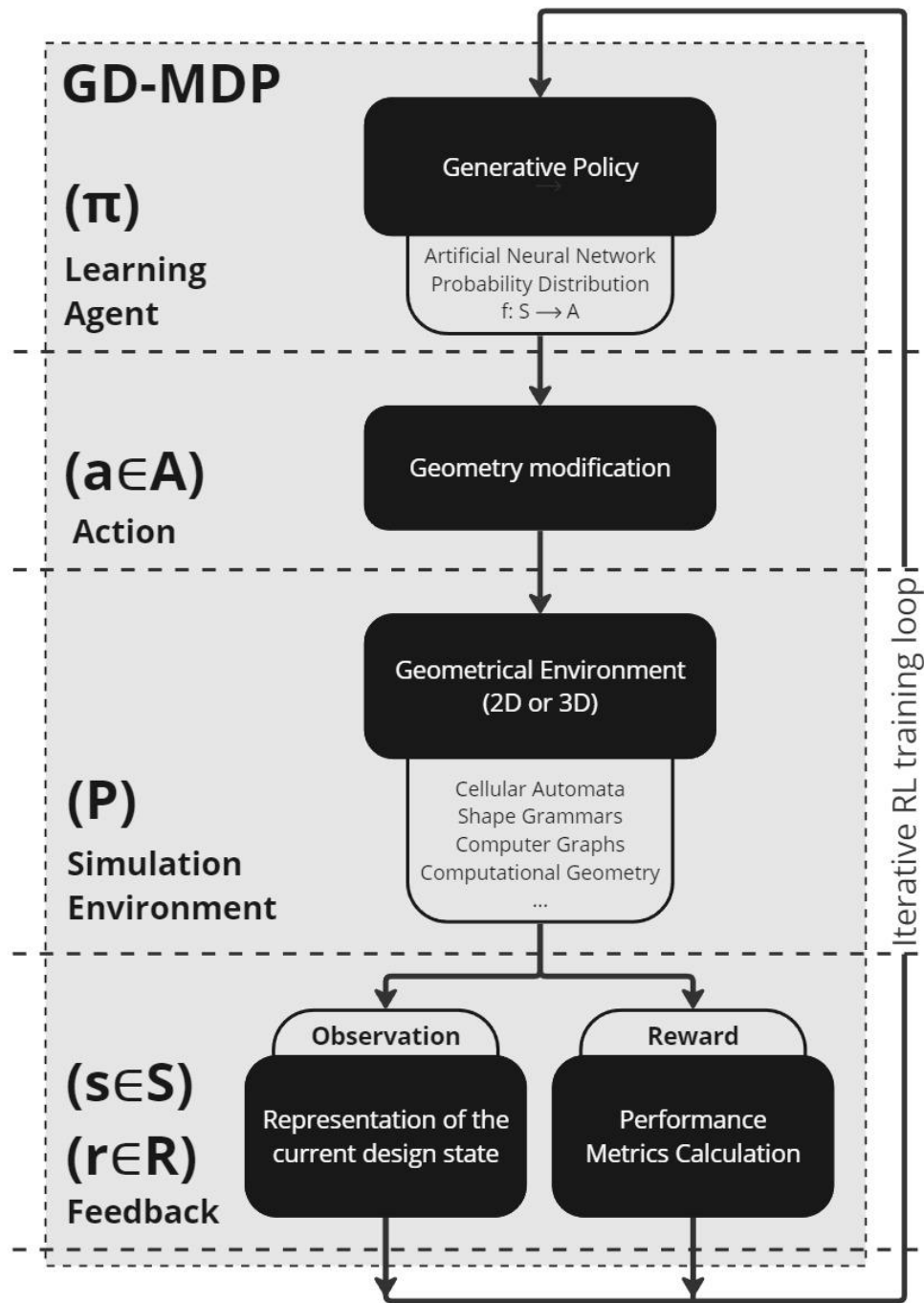


Figure 3.5 The generative design problem formulated as an MDP. The MDP could be solved according to a preset objective using various DRL methods. Produced by author.

configurations, and the agent would then take actions that represent incremental adjustments that would transition the model environment from one state to another, or from one design configuration to another (Figure 3.5). The actions within design MDPs would be a set of predefined spatial and geometrical modifications that the agent is allowed to perform at any point in time. As the agent acquires experience and learns the value of different design configurations and different action interventions given the current state of the design configuration, the agent starts developing a sequential policy of intervention that leads to better performing design configurations according to a clear design objective formulated as a reward function. The reward function could be a linear combination of various relevant performance metrics chosen by a designer or modeler with specific domain knowledge. Within this framework, the designer or modeler do not need to figure out the sequence of decisions or rules that would lead to better performing design solutions, they only need to know what constitutes a good or a bad design configuration and be able to formulate that mathematically in the form of a function that returns to the agent a numeric reward signal feedback after each performed alteration to the design proposal. Using DRL methods to solve this design MDP, this feedback would then direct the agent's learning of a policy or sequence of design decisions that yield the better performing design proposals with respect to the design objective (Figure 3.5).

Since the objective of any RL agent is to maximize cumulative reward, the RL agent within a generative design MDP (Figure 3.5) would learn sequences of actions that would maximize the linear combination of design performance metrics. Therefore, it would learn the sequence of modifications to the design configuration that would potentially result in design configurations with maximal design performance. The agent could also converge towards these better performing design options starting from any initial design configuration, no matter how good it is, since the learned design policy is conditional, meaning the actions chosen by the agent are adapted to the current design configuration that the environment is in. This learned design policy is then a general design improvement formula for that specific generative design problem.

### 3.3 Reinforcement learning applications in design

The use of reinforcement learning to solve design problems is very recent. There has been a surge in modelling design problems using MDPs during the past couple of years. DRL methods have proved to be effective in various domains of design ranging from structural design and topology optimization to design assembly, architectural spatial design, and urban design.

One of the first design fields in which the application of DRL methods were explored is structural engineering. Sun and Ma (2020) proposed four methods of structural topology optimization based on RL algorithms and deep computer vision techniques. Their proposed methods generated acceptable solutions for both 2D and 3D structural problems, including topology optimization of cantilever beams, and the optimization of the thickness of an atmospheric diving suit. Wibranek et al. (2021) found DRL algorithms useful in providing solutions to the adaptive reconfigurable structures problem. DRL agents have successfully been trained to sequentially assemble self-interlocking building block structures (Shih, 2016, as cited in Wibranek et al. 2021). This structural assembly system gets a geometric curve as input and tries to follow it to form structurally sound building block structures along the way. This method was tested on three different curves with increasing levels of complexity. The results were compared with evolutionary algorithm baselines. The DRL method was found to outperform the evolutionary algorithm in the most complex of experiments, namely the 3D curve case. DRL methods showcased a higher level of generalization while solving these sequential assembly problems. DRL methods were also used to optimize 2D truss structures (Ororbia & Warn, 2022). Ororbia and Warn (2022) propose an MDP where the initial state is a set of unconnected nodes that are connected sequentially to produce steel planar truss configurations that are adapted for various structural loads. Here again the researchers found that DRL methods were much more efficient in reaching optimal designs compared to genetic algorithm methods in this task. Huang (2021) implemented a reinforcement learning-based optimization system that designs shell structures composed of existing waste materials to contribute to research efforts towards producing structural assemblies that adopt circular design strategies in terms of constructure

material use. The assembly system proposed uses a combination of RL, computer vision techniques, and a search algorithm within a 3D environment in the game engine Unity3D to train agents to use pieces of waste constructure material and combine them in such a way that they constitute feasible shell structures. The DRL agent takes images of the state of the environment at each step and a reward signal representing the structural integrity of the produced shape at each step using the structural simulation Grasshopper plugin called Karamba3D (Preisinger, 2013). This method proposes a solution to a sequential decision assembly problem that would be complicated to solve manually in the real-world.

The MDP framework was also used within the domain of architectural design and spatial planning and configuration. Wang and Snooks (2021) explored the adoption of the MDP framework for generative design decision-making within a conceptual framework they named as a system of artificial design intuition. This research defined a theoretical framework that integrates RL methods with existing generative design procedures to tackle design problems. This framework was then tested within a practical experiment that aims at producing a voxel-based building massing that is structurally sound. The spatial logic used within this generative system was adapted from Le Corbusier's Domino system. The random walk algorithm was used within the environment in combination with the deep Q-learning for the learning and optimization of the generative policy. Veloso and Krishnamurti (2020) noted that most success in applications of RL methods to solving engineering problems are focused on non-architectural problems. They therefore proposed a framework to adapt these methods to spatial design problems. This research focused on multi-agent deep reinforcement learning (MADRL) methods to create setups where agents would interact to satisfy various architectural spatial requirements, i.e., surface area, adjacency, in producing interior space partitions and planar spatial configurations within grid-based environments. The agents trained were able to generalize to different initial settings where they constantly generate acceptable spatial configurations. Mintrone and Erioli (2021) also proposed a method using DRL algorithms in combination with the Wave Function Collapse (WFC) as a generative mechanism to produce architectural spaces that satisfy predefined objectives by the designer. This method went beyond the planar

organization of space and tested the DRL methods within 3D modular environments where the agent chooses from a set of spatial modules at each step to result in various 3D spatial configurations. However, to the contrary of the work proposed by Veloso and Krishnamurti (2020), the method proposed by Mintrone and Erioli was applied on a very simple MDPs with reduced state and actions spaces that do not reflect the complexity of real-world spatial planning problems.

Finally, the MDP framework was also found useful for urban design modeling and optimization, even though RL applications within this domain are relatively scarce within the literature compared to the other design fields. Song and Yuan (2021) proposed a DRL method to generate high-rise building massing to reduce the effect of lateral loads on the structure of the building within a grid-based environment. The



Figure 3.6 Sample of the sequential generation process of an urban block by a DRL agent. (Reprinted from Han et al., 2021, p.141)

effects of lateral loads were simulated within a wind tunnel and served as feedback for the DRL agent during the training process. The proposed DRL method was found superior to traditional generative design computational methods in terms of efficiency in design space exploration and reproducibility of the solution since the learned generative policy could be used post-training to produce adapted masses of buildings in similar contexts. However, once again, the grid environment used within these experiments was small to limit the complexity and scale of the MDP the DRL agent had to solve in order to reduce the computational and temporal cost of the experiment.

DRL methods were also used to solve the sequential decision problem of urban block generation (Han et al. 2021), in an application that is the closest to the task tackled within this thesis. Han et al. (2021) proposed a performance-based generative design method to produce urban block massing to maximize thermal comfort based on factors such as direct sunlight hours, solar heat gains. The DRL agent within this method was based on computer vision models to be able to develop an encoding of the state of the environment fed to it at each step in the form of an image tensor. The method was tested on a case study of an urban renewal project in Shenyang, Liaoning Province, China, where the DRL agent successfully proposed urban block schemes that performed on an acceptable level in terms of outdoor thermal comfort. The DRL agent arranged one building in the site at a time in a training episode by choosing its positional x and y coordinates, as well as the length, width, and height of the building mass. After about 150 hours of training, the proposed method generated 2179 satisfactory design solutions. The generated proposal that received the highest reward was then manually modified by the authors to provide the final proposal (Figure 3.6). The spatial organization and circulation between buildings were however considered only indirectly within this model and the focus was mainly on the thermal performance. The DRL agent was only directed towards producing more orthogonal circulation configurations through the Hough transform function feedback within the reward signal. This feedback measures how aligned building masses are relative to each other, the more aligned the building edges are the higher was the reward.

## CHAPTER 4

### 4 Experiments and Results

Based on the RL background reviewed in the previous chapter, and its recent application to generative design problems, this chapter proposes a framework to model and simulate self-organized settlement dynamics using single-agent DRL methods. The proposed framework is then used to formulate two self-organized settlement models, namely the ‘Beady Ring’ model (Hillier & Hanson, 1984) and the ‘Path Node’ model (Erickson & Lloyd-Jones, 1997), as MDPs. RL agents are then trained to produce settlements by maximizing different reward signals within these environments.

#### 4.1 Self-organized settlement morphogenesis as a Markov decision process

Previous studies mentioned in chapter 3 have tested the MDP framework and found it to be useful and effective for many generative design tasks. Self-organized settlement modeling, and more broadly urban form modeling, could also benefit from many advantages that this modeling framework has to offer, most notable of which are the following:

- First, the MDP framework enables digital modeling of urban growth **without the specification by the human modeler of the exact sequence of algorithmic rules that would result in the sought after settlement form**. The modeler only needs to specify what represents favorable urban performance for each specific settlement modeling and optimization task through the definition of an exact

mathematical reward function.

- Furthermore, representing the self-organized settlement growth modeling problem as a MDP offers the possibility to deploy various RL methods in the **optimization of the settlement growth towards a predefined objective**. RL methods have been found to be more sample-efficient than genetic algorithms which are widely used within the generative design field presently, as discussed in section 3.2 of this thesis.
- Finally, the MDP framework closely reflects the actual dynamics of self-organized settlement morphogenesis where **individuals and various types of actors perform interventions locally within the settlement environment and adapt their policy of action to the positive or negative feedback that they receive**, resulting in the continuously adapted settlement form to the socio-economic context and the environmental constraints, as covered within chapter 2 of the thesis.

Based on the fact that self-organized settlement morphogenesis is a sequential decision-making aggregation procedure, as established within literature discussed in chapter 2, this thesis proposes a conceptual framework for self-organized settlement growth modeling based on the MDP theory, that could be solved using DRL methods. This proposed framework is referred to hereinafter as the **self-organized settlement Markov decision process (SOS-MDP)** (Figure 4.1).

As previously stated within section 3.1.3, an MDP is defined by the tuple  $(S, A, \mathcal{P}, \mathcal{R}, \gamma)$  where  $S$  is the state space,  $A$  is the action space,  $\mathcal{P}$  provides the transition dynamics of the environment,  $\mathcal{R}$  is the set of all possible rewards, and  $\gamma \in [0,1]$  is a discount of future reward needed for the estimation of cumulative future reward starting

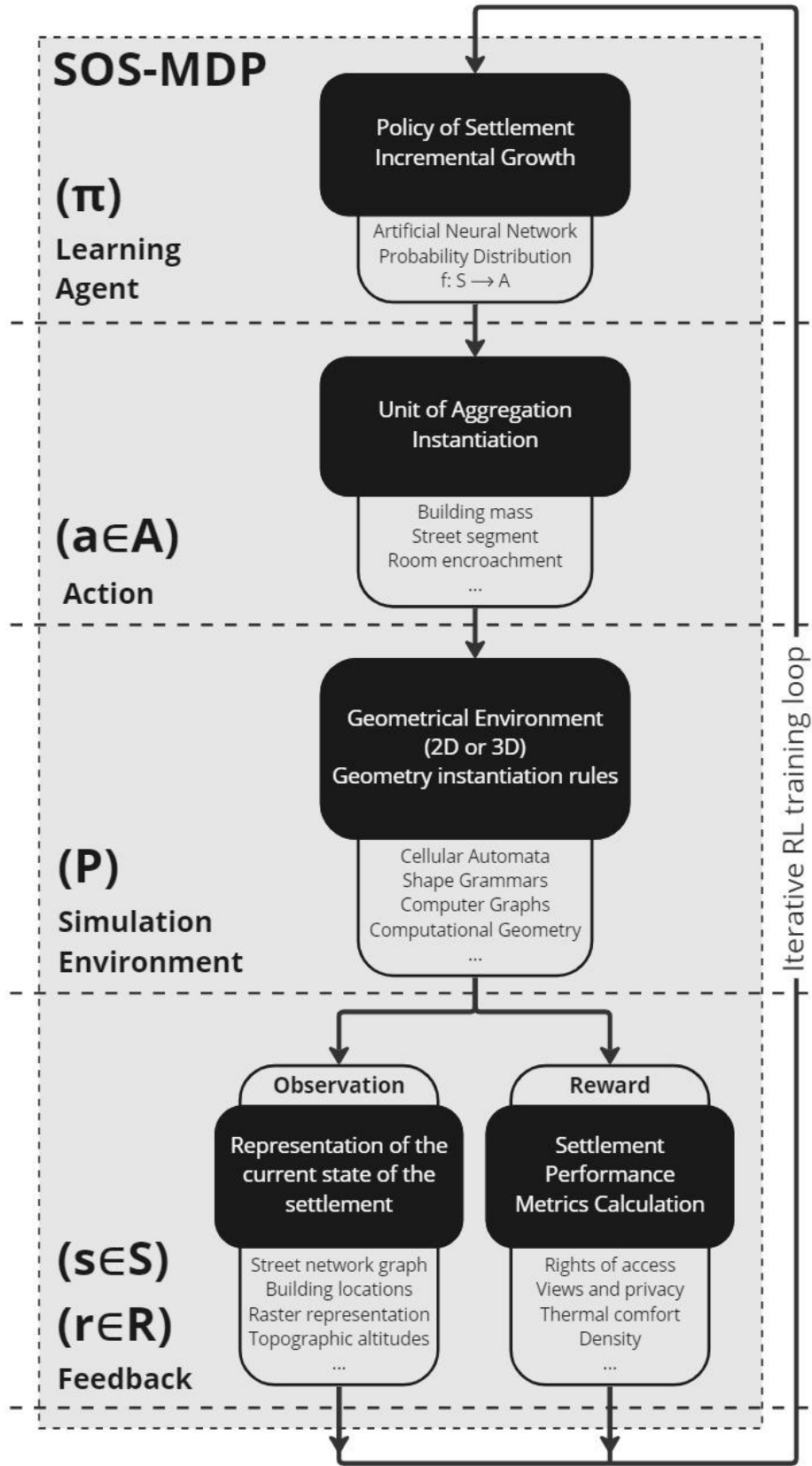


Figure 4.1 SOS-MDP framework diagram. All the necessary components to define an MDP are defined for the self-organized settlement case. The MDP could be solved according to a preset objective using various DRL methods. Produced by author.

from the current state and knowing that the agent chose a certain action. Added to this tuple is the policy  $\pi$  which enables the agent to act within the environment depending on its current state and is the component that is optimized using RL methods and is considered the solution to the MDP. All these components need to be defined in the context of self-organized settlement morphogenesis simulation to constitute the proposed SOS-MDP (Figure 4.1) that could be then used as a general framework to leverage DRL methods in self-organized settlement generation and modeling.

The ‘state space  $S$ ’ within SOS-MDPs corresponds to all the possible settlement spatial configurations that the environment could transition to at any given time. The ‘state observation  $s$ ’ could take the form of a vector representation of the positions and orientations of all neighboring buildings, it could be accompanied by a graph representation of the street network within the settlement, or it could even be a raster image representation of the current state of the settlement. The observation must be a compact numeric representation of the morphological state of the settlement that could be processed by the ‘policy function  $\pi$ ’ at each step to determine the subsequent action. The state representation within the SOS-MDP should satisfy the Markov property. A representation of the positions of the various building masses and the street network within the settlement at any one time is sufficient for an agent to take an aggregation action to further the growth of the settlement. The history of all actions taken that led to the current state of the settlement is not required to perform a well-informed future sequence of settlement spatial aggregation or intervention that would improve the current state of the settlement.

The ‘action space  $A$ ’ corresponds to all the aggregation actions that the agent could possibly take at any given time to alter the state of the settlement configuration. These actions would represent the instantiation of an additional unit of aggregation within the environment. This could take the form of a vector of parameters determining the location, dimensions and orientation of a new building mass, or the addition of a new street segment, or other forms of spatial aggregation through encroachment like adding new rooms to already existing buildings within the settlement.

The ‘probability of transition from one state to another  $\mathcal{P}$ ’ is defined by the rules

that govern the geometric transformations that happen within the environment as a consequence of the agent's actions. The model of the environment might employ any of the computational spatial representation methods such as cellular automata and could involve shape grammars and various computational geometry transformations as part of the dynamics of the environment. In the context of settlement growth modeling, the 'discount factor  $\gamma$ ' is set relatively close to 1 if the modeler is only interested in the final urban form of the settlement since in that case future cumulative rewards would be more important than any immediate reward that an agent could receive at any given time. The discount factor could be set closer to 0, however, if the modeler is interested in simulating an urban growth environment where the agent is greedy and gives little regard to future expected reward.

Finally, the reward function  $f: S, A \rightarrow \mathcal{R}$  is crucial for the learning of the agent, and therefore important for the determination of the aggregation policy that results in the settlements form, as it determines the objective of the settlement's optimization process. This function must be well formulated in such a way that it provides a feedback signal that rewards actions that led to favorable urban performance outcomes and penalizes actions that deteriorated this performance. The determination of what is desirable or not in terms of urban performance could be subjective and is up to the modeler to specify and could change from one SOS-MDP to another. The reward function could take the form of a linear combination between various morphological, socio-economic, and/or environmental computed metrics resulting in one numeric reward value sent back to the agent at each step.

Next, the SOS-MDP framework is used to adapt two self-organized aggregation models, namely the 'Beady Ring' model (Hillier & Hanson, 1984) and the 'Path Node Generator' model (Erickson & Lloyd-Jones, 1997), into RL environments where learning agents are trained to develop policies that maximize the performance of the urban form in each of the environments.

## 4.2 Experiment Setting

To demonstrate the use of the proposed SOS-MDP framework in self-organized settlement modeling, three MDPs are created based on the framework and then solved. This section entails the details of three RL environments developed based on the dynamics of the ‘Beady Ring’ model (Hillier & Hanson, 1984) and the ‘Path Node Generator’ model (Erickson & Lloyd-Jones, 1997). The three environments are referred to as the ‘Generalized Beady Ring’ (GBR) environment with two versions with different reward functions, and the ‘Generalized Path Node’ (GPN) environment. These environments expand the range of possible urban configurations that the original models could produce by allowing the agents to freely choose from a set of possible actions at each step of aggregation, as opposed to having preset probability distributions over those choices like in the original implementations of these models, as previously discussed in section 2.2.2.

### 4.2.1 The Generalized Beady Ring Environment version 1

The first environment, the GBR version 1 environment, is a cellular automata grid world environment based on the ‘Beady Ring’ model (Hillier & Hanson, 1984). The grid is one of 36 cells in length and width, where each cell could take one of three different states at a time; (i) a carrier cell represented in grey, (ii) a building cell in black, and (iii) a street or open urban space cell represented in white (Figure 4.3). At each timestep, the agent decides whether the current cell would be a building or a street cell. If the agent decides to place a street cell, then the adjacent cells that form its von Neumann neighborhood (Figure 4.2) are added to the list of cells where future actions would take place. This process leads to the aggregative sequential generation of

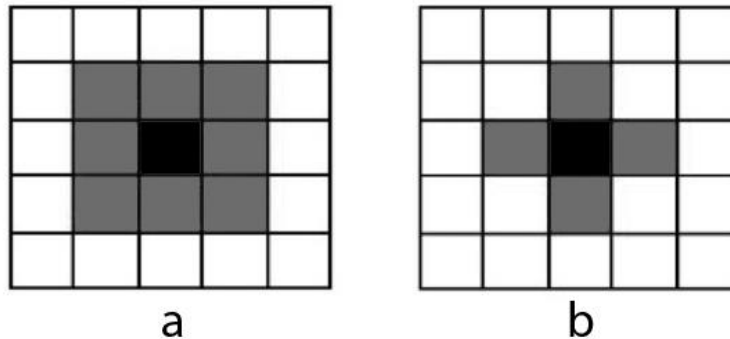


Figure 4.2 The Moore neighborhood (a) and the von Neumann neighborhood (b). Produced by author.

settlement configurations where all the building cells have access to one street cell at least, maintaining the right-of-way. This is a hard constraint in the form of a terminal state condition. The episode terminates when (i) there are no carrier cells adjacent to street cells left for an action to take place, or (ii) if all the cells in the grid world are already visited.

To take these decisions, the agent receives information about the current state of the grid world environment. The agent gets either a global observation in the form of an image representing the whole grid world, or a local observation in the form of an array with the states of the cells under the Moore neighborhood of the current action cell (Figure 4.3). The extent of this Moore neighborhood observation is adjustable. A discussion of the effect of the size of the observation on the RL agent's learning is found later in this chapter.

The objective of RL agents within this environment is to grow the settlement as much as possible while maintaining right-of-way to all the building cells. The agent

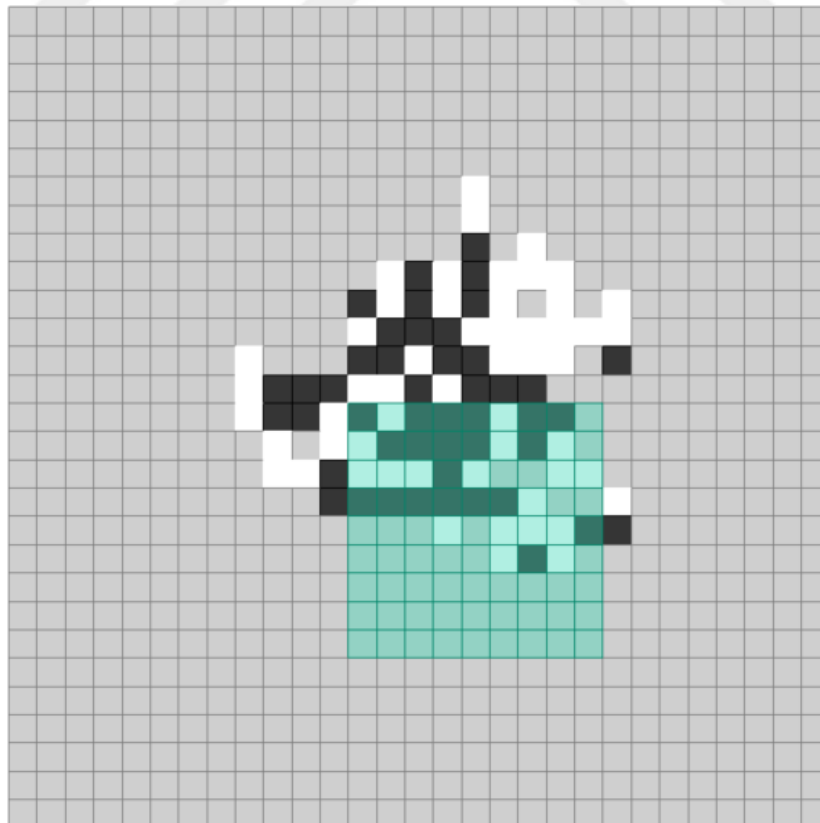


Figure 4.3 The GBR environment with local observation, represented by the green area. Produced by author.

receives a +1 reward signal for every building it instantiates as an incentive to continue growing the settlement further.

#### 4.2.2 The Generalized Beady Ring Environment version 2

The second environment, the GBR version 2 environment, is similar to the first environment. They have the same dynamics, the only difference being in the reward signal. In this second environment. The agent not only aims to grow the settlement as much as possible while maintaining right-of-way, but it also seeks to maximize the number of urban blocks within the settlement. To incentivize the agent to do so, an additional +10 reward signal is received by the agent each time an urban block of six or more building cells is formed. The urban blocks are detected at each timestep using the Hoshen-Kopelman algorithm (Hoshen & Kopelman, 1976), which is a computationally efficient algorithm used to label clusters of cells on a grid (Figure 4.4). Each timestep, if a new building cell cluster is detected and it contains six or more cells, the agent is reward with a +10 in addition to the +1 reward it receives for instantiating a building as an incentive to keep growing the settlement. The results of training RL agents on both versions of the GBR environment would show the effect of changing the reward signal on the sequential policy of action that the agent learns.

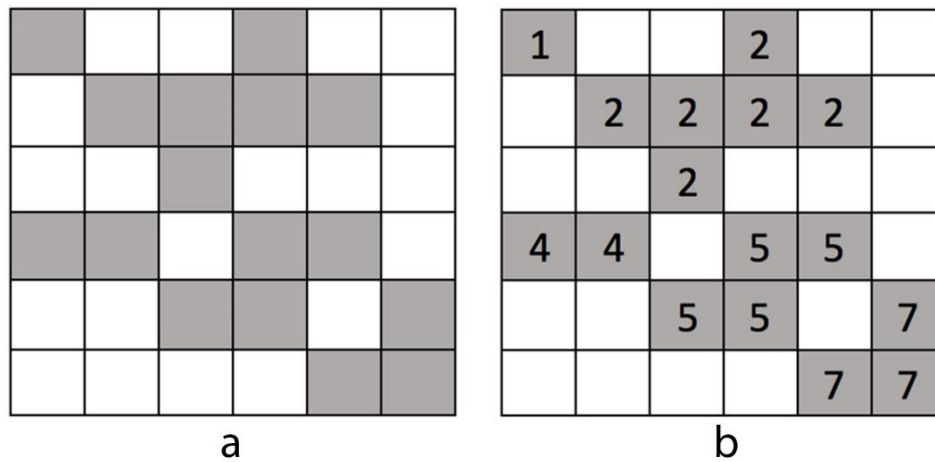


Figure 4.4 A cellular automata grid (a) and its labeled clusters using the Hoshen-Kopelman algorithm. In this example, the agent receives a reward only for cluster number 2, since it has six cells. Produced by author.

### 4.2.3 The Generalized Path Node Environment

The third and last environment, the GPN environment, introduces more complexity relative to the GBR environments. It is based on the ‘Path Node Generator’ model by Erickson and Lloyd-Jones (1997), which built upon the ‘Beady Ring’ model and introduced more flexibility to the cellular automata settlement representation to better fit the dynamics of settlement morphogenesis at the local scale of unit of aggregation, as previously reviewed in section 2.2.2. The cells within the GPN environment do not conform to a grid, rather, the agent sequentially instantiates cells starting from an initial position. The agent chooses the state of the current cell (building or street cell), as well as its orientation from a discrete set of orientation angles with a minimum of  $-10^\circ$  and a maximum of  $10^\circ$  and increments of  $5^\circ$ . This makes the action space larger but offers the agent the possibility of generating more convoluted street networks and urban morphologies. Similar to the GBR environment setup, if the current action cell is a street, then the adjacent cells forming its von Neumann neighborhood are added to a list of potential action cells (Figure 4.2).

This environment also introduces topographic constraints to settlement growth. The episode terminates in the occurrence of one of the following three conditions: (i)

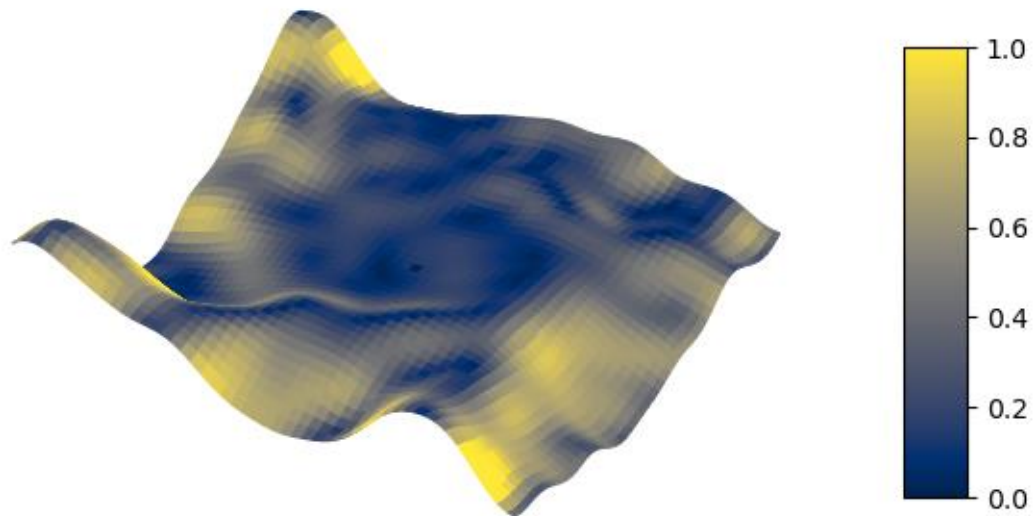


*Figure 4.5 A sample topography generated using the Perlin Simplex noise algorithm, with slope values normalized between 0 and 1. Produced by author.*

when there are no potential action cells adjacent to street cells left, or (ii) if a cell is placed outside a predefined boundary, or (iii) if a street cell is placed in an area where the topography slope is higher than a certain threshold (Figure 4.5). A threshold of 0.3 is adopted for the experiment results discussed in the following section of this chapter. The simulated topography conditions are generated using the Perlin Simplex noise algorithm (Perlin, 2001). It is a type of gradient noise method that has been widely used for procedural terrain generation. The generated topography is in the form of a 100\*100 2D array of elevation values, which is used to calculate an array of the same size containing normalized terrain slope values (Figure 4.5), which is then used to set the topographic hard constraint within the environment. The agent still receives a +1 reward signal every time it instantiates a new building cell. The objective of the agent within the GPN environment is to grow the settlement as much as possible while maintaining the right-of-way as well as adapting the urban form to avoid high slope terrain.

To learn its conditional policy within this environment, the agent gets information about the states, location in the form of x and y coordinates, and terrain slope values of the last 10 visited cells. The observation space could also be expanded to include more than 10 previously visited cells within the GPN environment implementation proposed in this thesis, however, it is limited to 10 within the experiments due to computational budget considerations. So, the agent receives a 40 long vector of values normalized between 0 and 1 as an observation at each timestep.

### 4.3 Results and Discussion

After transforming the ‘Beady Ring’ and ‘Path Node Generator’ models into MDPs following the framework proposed in section 4.1, DRL agents are now trained within these MDPs to maximize their reward signal and therefore learn settlement growth policies that improve the settlement’s performance overtime. The agents are trained using two RL algorithms, namely the DQN algorithm (Mnih et al., 2013), a value-based approach, and the PPO algorithm (Schulman et al., 2017), a policy-based approach. After 3 days of hyperparameter tuning of each of the two algorithms on each environment, a process where the parameters of the RL algorithm such as learning rate are fitted to the specific task at hand, each agent is trained for 3 million timesteps.

Within the GBR version 1 environment, the objective is to grow the settlement as much as possible while maintaining right-of-way. Given a local observation only in the form of the states of the Moore neighborhood (Figure 4.2) cells, the DQN agent learns to grow the settlement up to 60 buildings on average, and a maximum of around 200 buildings (Figure 4.6). The PPO agent, however, grows the settlement up to 450 buildings on average with the best episodes yielding settlements of 700 buildings (Figure 4.6), therefore clearly outperforming the DQN agent. The PPO agent achieves

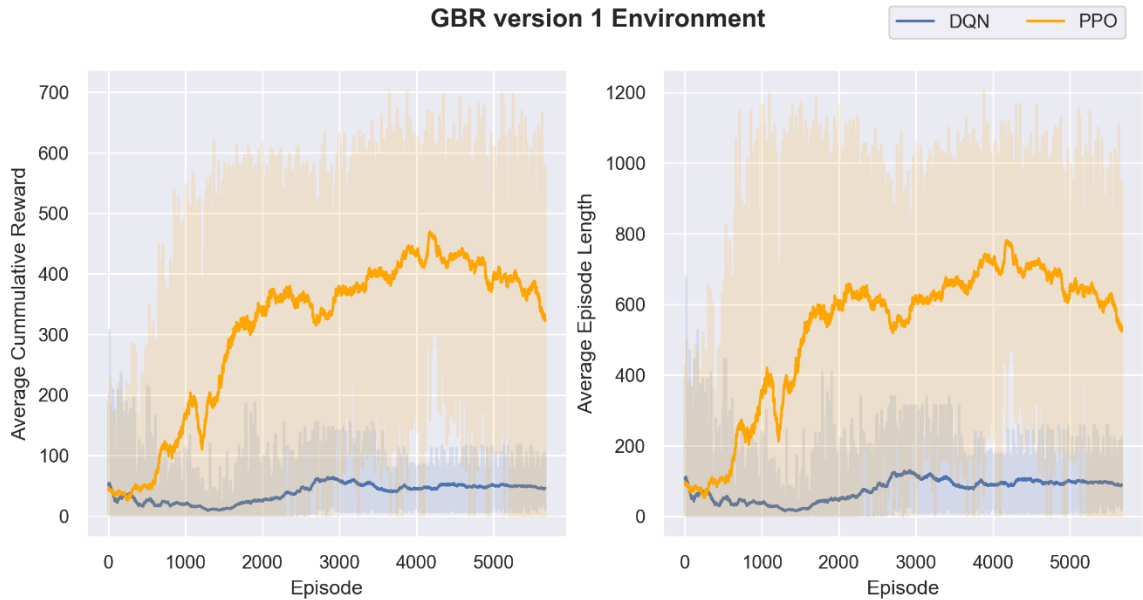
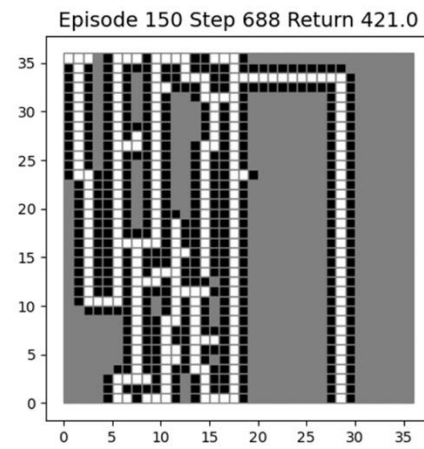
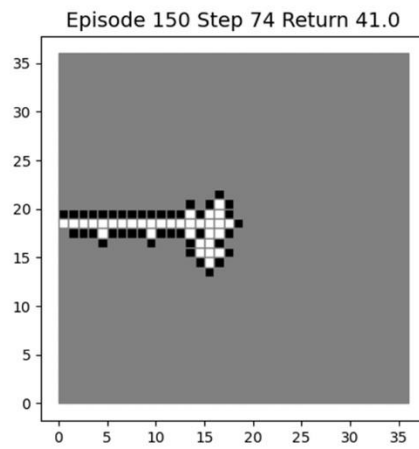
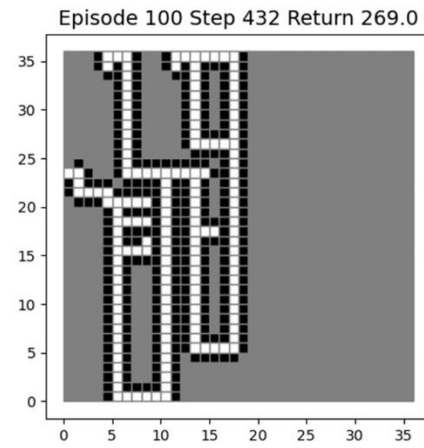
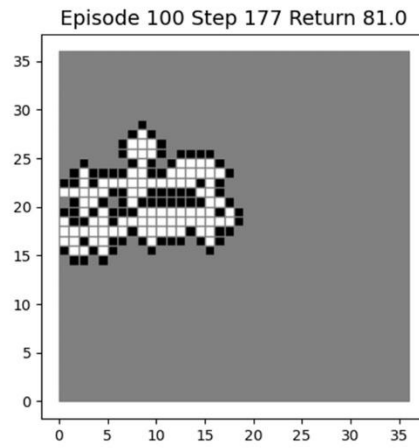
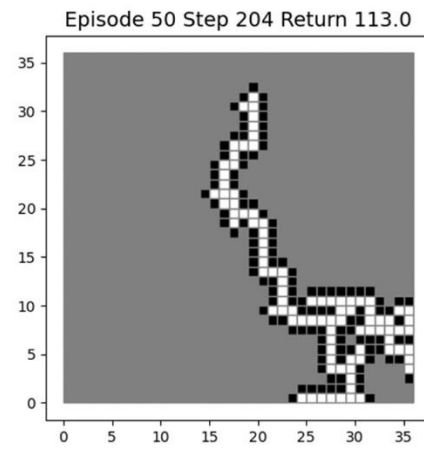
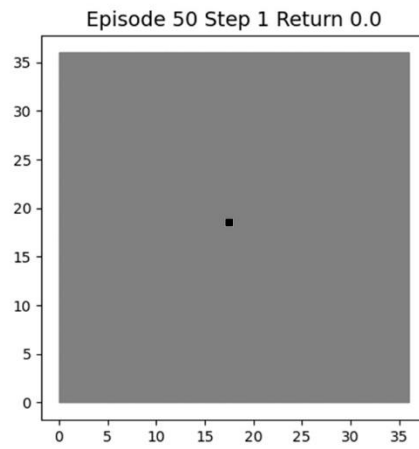


Figure 4.6 Training results on the GBR version 1 environment using the DQN and PPO algorithms. Produced by author.



DQN

PPO

Figure 4.7 Renders of the terminal states of the training episodes of the GBR version 1 environment using both the DQN and PPO algorithms. The renders are taken every 50 episodes with 32 parallel processes during training. Produced by author.

connected streets with buildings on each side (Figure 4.7). The DQN agent, however, only learned to place many street cells adjacent to each other (Figure 4.7) to prolong the episode length (Figure 4.6), and therefore increase its chances of instantiating a building cell without resulting in a terminal state, and thus slightly increasing the reward signal. PPO training is not sample efficient enough to converge to an optimal solution within the 3 million timesteps of training within this task, but the PPO agent quickly managed to learn a conditional policy based on a local observation to clearly improve the performance of the settlement within this environment.

The size of the observation that the agent receives at every timestep has an impact on its learning and performance. To demonstrate this impact, the DQN agent is trained with both local observation in the form of the states of the Moore neighborhood (Figure 4.2), as well as a global observation in the form of an image of the global state of the cellular automate grid world, this latter observation is handled using a convolutional neural network (CNN) architecture. The DQN agent with a local observation starts improving its performance around episode 1500 and maintains a clear advantage in performance over the global observation DQN agent (Figure 4.8). This is due to the much larger observation space that the global DQN agent must explore to start learning any meaningful strategies. A larger observation space exponentially grows the space that the agent must explore, slows the learning process, and makes the agent prone

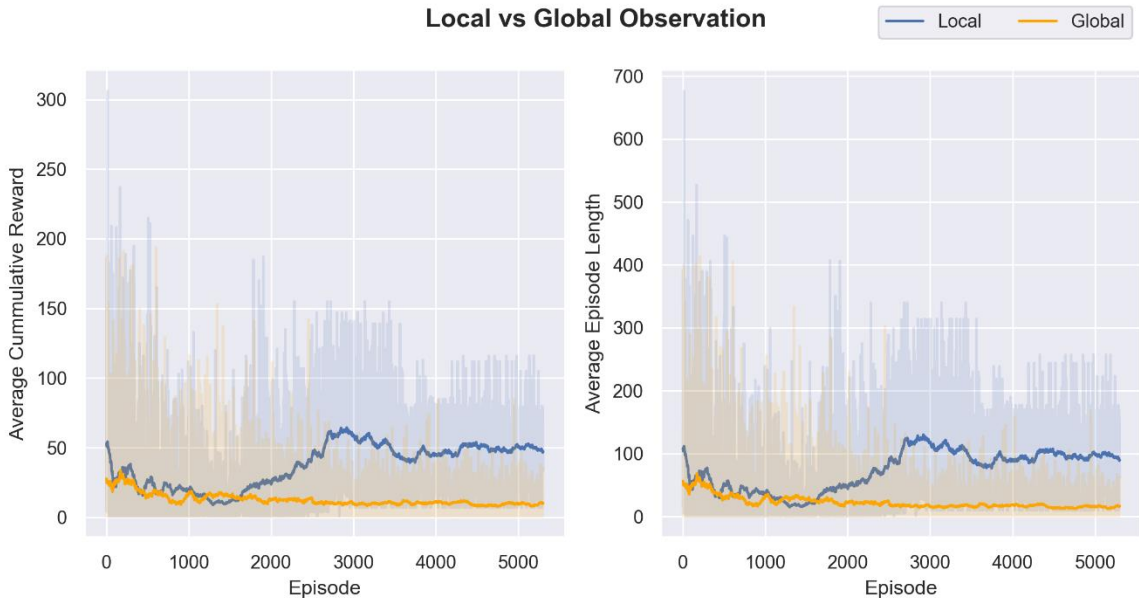


Figure 4.8 Training results on the GBR version 1 environment using the DQN algorithm with both global and local observation. Produced by author.

to converge to suboptimal policies. The global DQN agent started exploring less relative to the local DQN agent, as its recorded episode lengths started collapsing from episode 3000 onwards (Figure 4.8). Additionally, the local Moore observation contained enough information for the PPO agent to learn effective policies to grow the settlement, and so the local observation with a PPO agent is used within the GBR version 2 environment.

The reward function is expanded in the GBR version 2 environment to showcase the impact of reward signal on the policy learned, and thus the generated settlement's form. The PPO agent is encouraged to form urban blocks, with an additional +10 reward, as well as grow the settlement while maintaining the right-of-way within this environment. Similar to the results in the GBR version 1 environment, the PPO agent quickly learns to allocate street cells linearly as street segments and then incrementally place building cells on both sides of those street segments. Most of the rapid improvement happens within the first 2000 episodes (Figure 4.9), then within the following 5000 episodes to training the agent just explores different settlement configurations without significant improvement. The episode length reaches around 1200 steps (Figure 4.9), which is close to the maximum episode length of 1296 steps possible within this environment. The PPO agent, therefore managed to grow the settlement almost to the full capacity of the grid world.

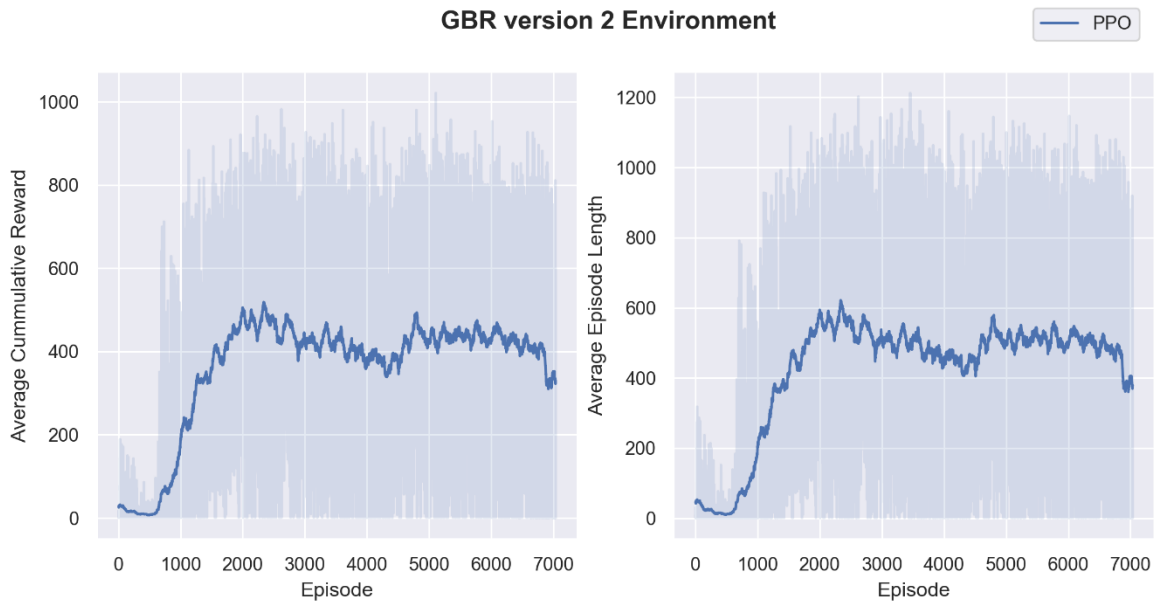


Figure 4.9 Training results on the GBR version 2 environment using the PPO algorithm. Produced by author.

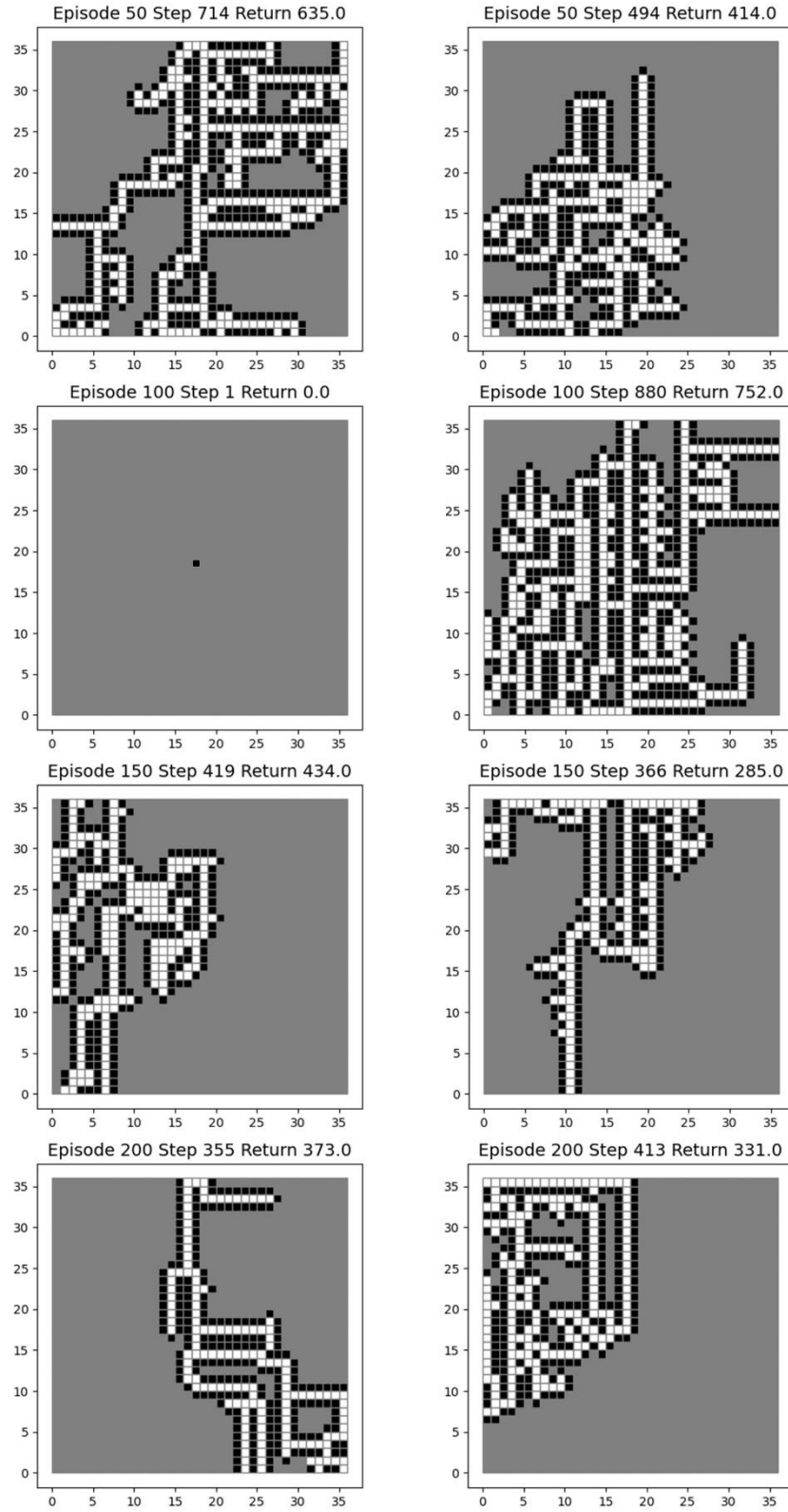


Figure 4.10 Renders of the terminal states of the training episodes from 2 parallel instances of the GBR version 2 environment using PPO algorithm. The renders are taken every 50 episodes with 32 parallel processes during training. Produced by author.

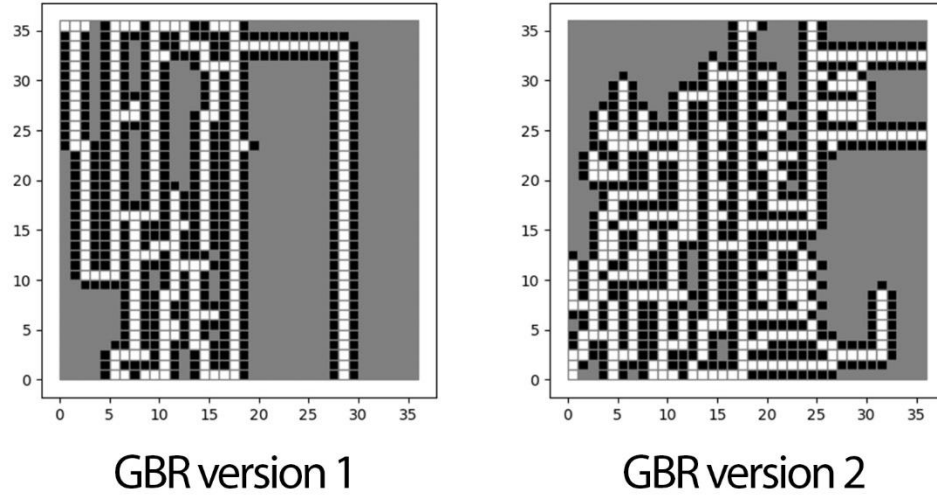


Figure 4.11 Sample settlement configurations generated using PPO within both versions of the GBR environment, showcasing the effect of the reward signal on the policy learned by the agent. Produced by author.

More importantly, the PPO agent also managed to reach around a 1000 in cumulative episodic reward, which is much higher than its maximum of 700 within the GBR version 1 environment, by successfully managing to create urban blocks within the settlement (Figure 4.11). These results showcase the importance of the reward signal while using RL methods for self-organized settlement modeling, and for generative design tasks in general. Using the same algorithm and a similar environment setup with the only difference being the reward signal, the agent learns to adapt its policy to maximize the reward, resulting in a different morphology that maximizes the predefined performance metric each time.

Finally, within the GPN environment, the PPO agent rapidly learns to reach settlements with around 60 buildings within the first 1000 episodes of training and then keeps improving performance gradually after that until reaching an average settlement size of around 100 buildings, and a maximum settlement size of around 140 within the last few episodes of training (Figure 4.13). The PPO agent does so while maintaining the right-of-way and adapting the settlement form to the topographic condition by learning to not place street segments within sloppy terrain (Figure 4.12). The DQN agent on the other hand explores various actions within the first 1000 episodes, generating settlements with 20 buildings on average, with the largest settlements generated within

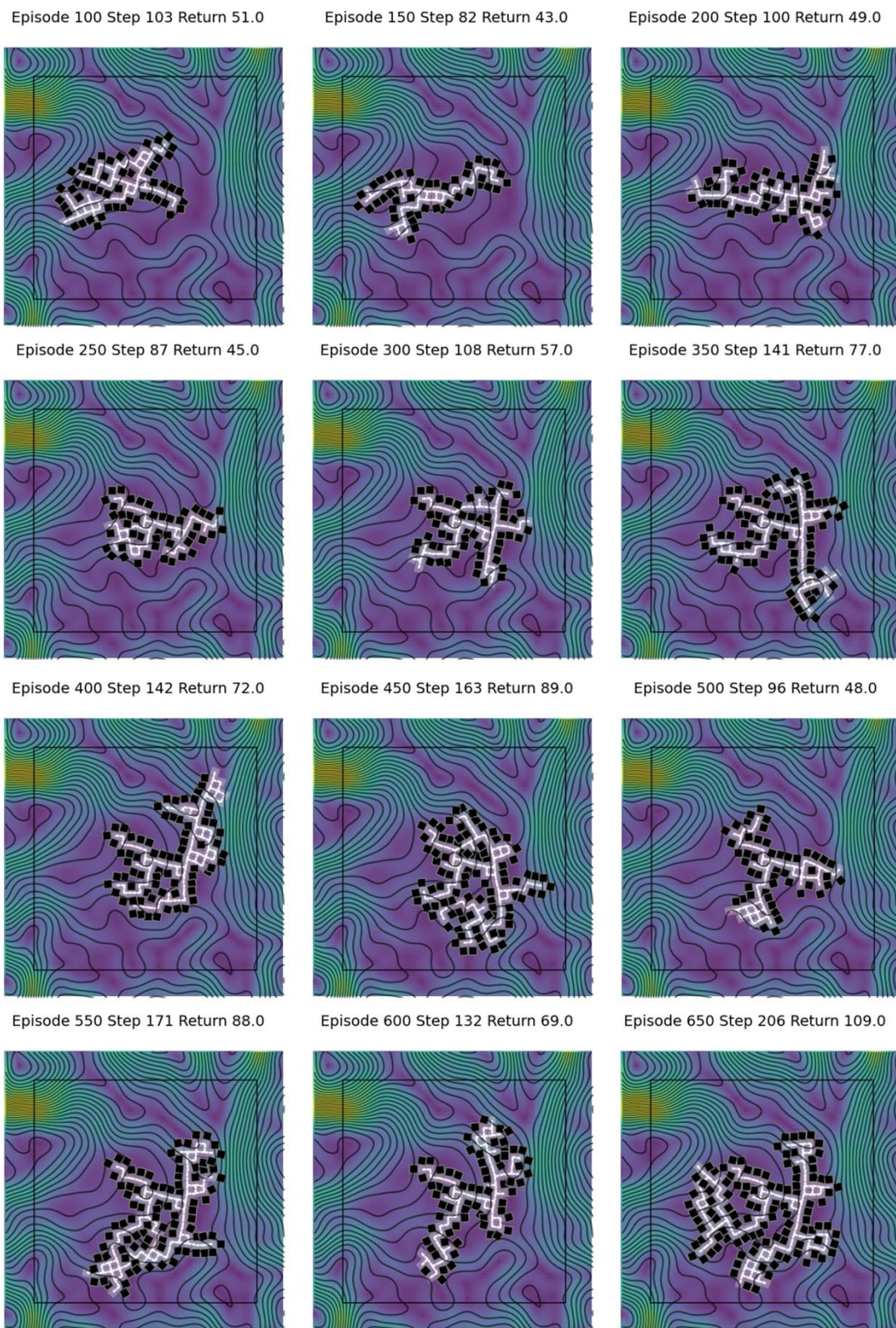


Figure 4.12 Renders of the terminal states of the training episodes of the GPN environment using PPO. The renders are taken every 50 episodes with 32 parallel processes during training. Produced by author.

first 1000 episodes being of around 65 buildings. The DQN agent fails to learn any successful policy to grow the settlement further, with the agent reaching a terminal state almost immediately after the start of each episode (Figure 4.13). This could be due to the algorithm's sensitivity to its hyperparameters or the initialization parameters of the model. Again, the PPO policy-based agent clearly outperformed the DQN value-based agent, even though it is not sufficiently sample efficient to converge to optimal policies within the 3 million timesteps allocated to training.

The PPO agent was further tested on different topographic conditions. The PPO agent was trained on three different terrains: (i) an introvert terrain condition with many valleys, (ii) a linear terrain formation with narrow ridges, and (iii) an extrovert terrain representing a hilltop (Figure 4.14). The RL agent, in each case, autonomously learns a sequential policy of aggregation to grow the settlement in adaptation to the topographic condition. The PPO agent does not reach the optimal policy for any of the terrains due to the limitation of the training budget allocated. These experiments demonstrate the capacity of RL algorithms at learning self-organized settlement aggregation policies adapted to various environmental constraints while maximizing performance, given an MDP formulation of the bottom-up urban growth process.

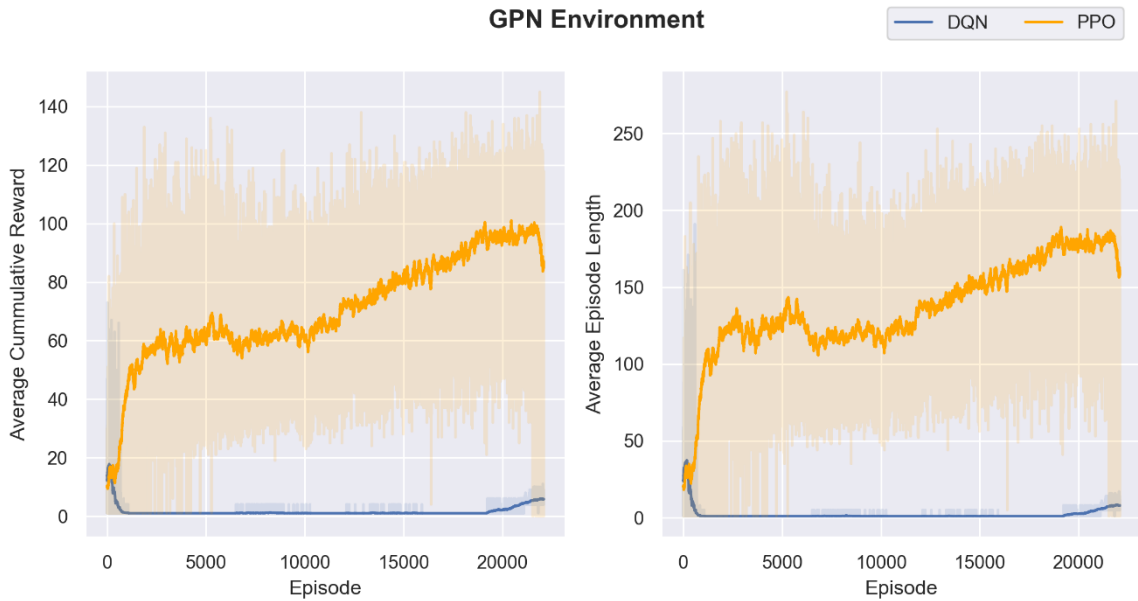


Figure 4.13 Training results on the GPN environment using the DQN and PPO algorithms. Produced by author.

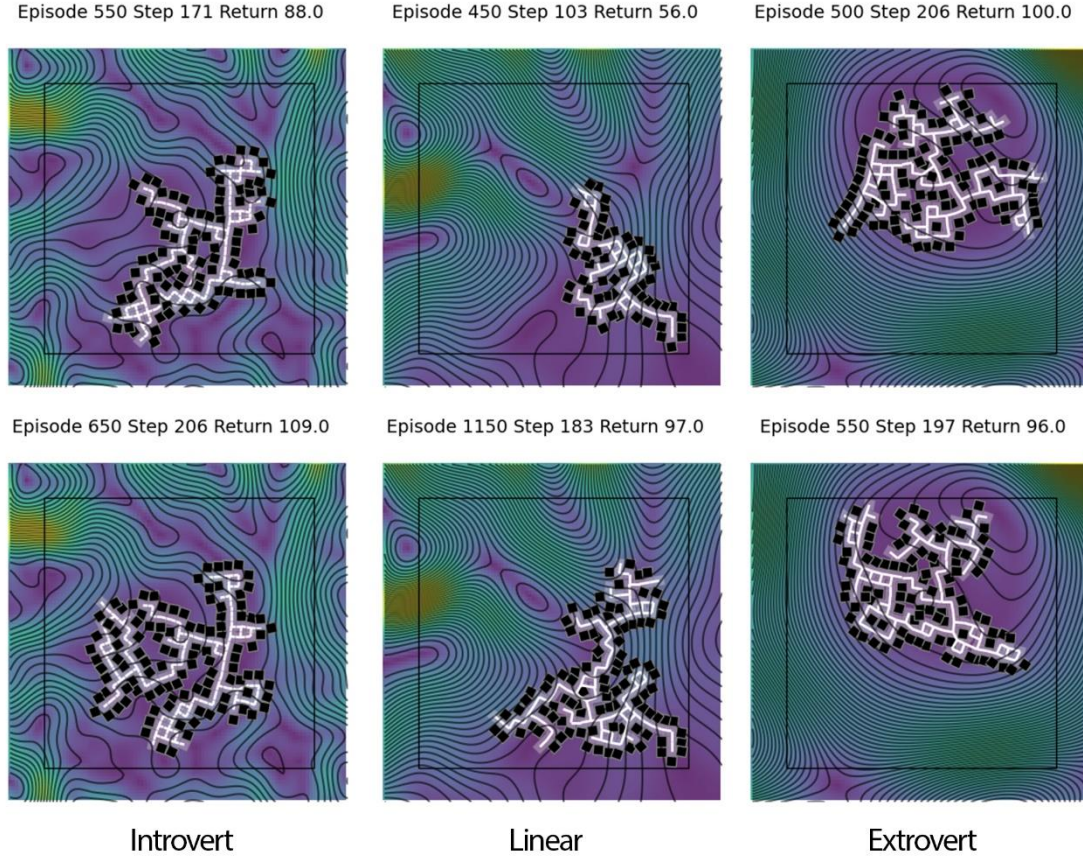


Figure 4.14 Renders of the terminal states of the training episodes of the GPN environment with different topographic constraints (introvert, linear, and extrovert) using PPO. Produced by author.

The proposed method makes a valuable contribution to address common challenges in urban growth modeling by automatically learning self-organized settlement generative policies based solely on a predefined feedback signal, as opposed to having to explicitly formulate the algorithmic rules of aggregation at each timestep to generate or model a specific settlement form using rule-based methods. This method remains limited in terms of its applicability to larger, more realistic settlement growth models, owing to the high computational efficiency required in the environment implementation and the insufficient sample efficiency of DRL algorithms, which necessitates many millions of timesteps to learn meaningful policies. However, it holds many advantages relative to existing methods and a strong applicability potential in augmenting the capacities of designers and planners in generating, analyzing, and running projections of urban form growth into the future:

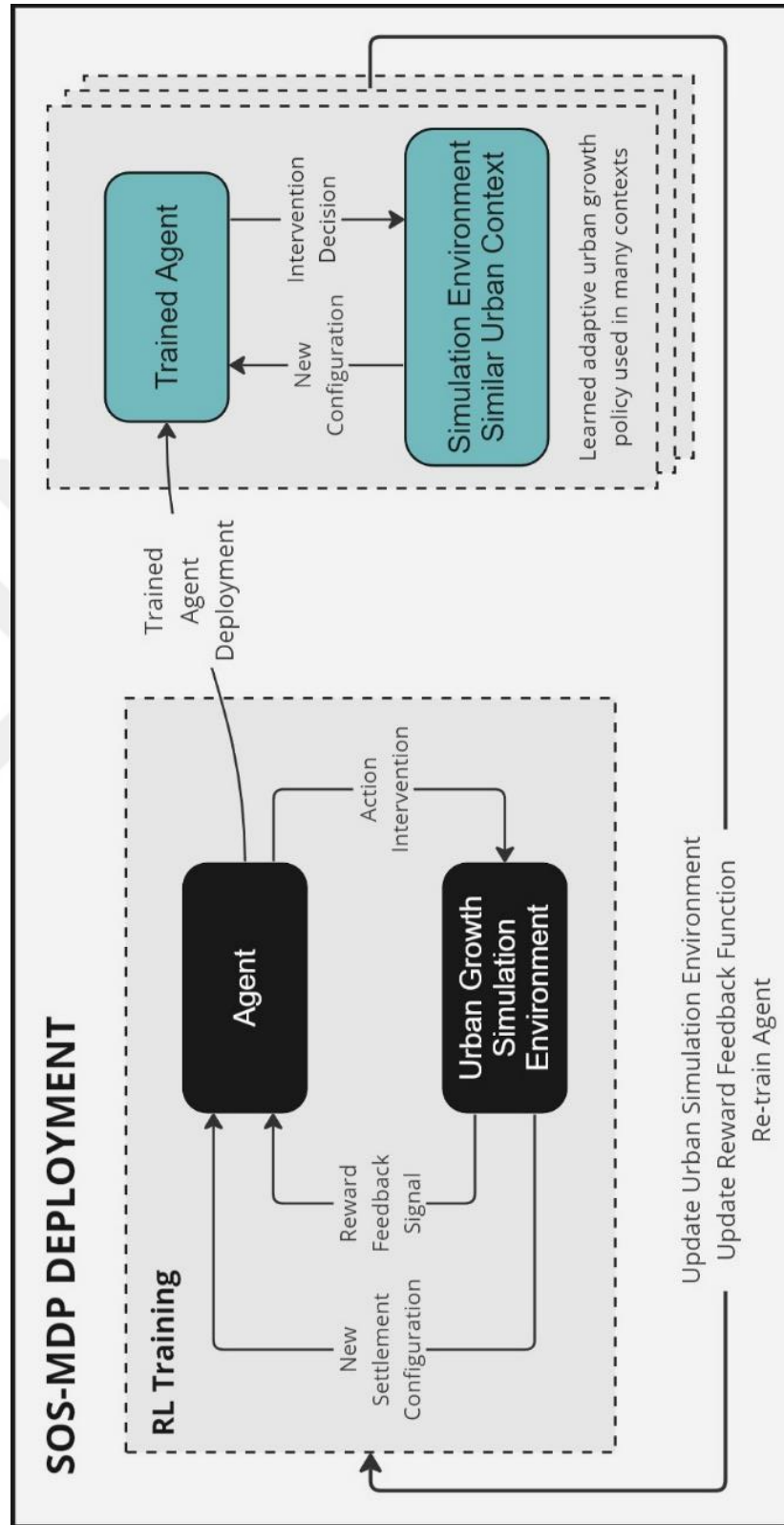


Figure 4.15 The process of training and then deploying trained agents on similar urban growth simulation environments. Produced by author.

- The proposed method **encodes the generative decision policy as a function that can be reused** on similar settlement generative conditions and contexts in the future (Figure 4.15); previously proposed methods that are not learning-based must rely on performing the rather computationally expensive optimization from scratch each time.
- The proposed learning-based method encodes the policy as learned parameters of a neural network or a function that maps the state space to the action space, it can be used to **simulate settlement growth under other similar conditions without requiring the agent to be retrained** (Figure 4.15). This justifies the increased computational cost of training these RL agents as they could be reused to produce many sequences of generative decisions instantaneously within a multitude of similar contexts.
- Given a predefined performance objective, the proposed method could be used to **forecast future settlement growth**.
- Finally, even though, the encoded policy is black box neural network that is hard to translate into explicit generative rules, planners could analyze the aggregation sequences produced by these learned policies to **find out the exact sequence of aggregation** that the settlement should follow to converge into an **urban form that achieves a certain desired performance**.

## **CHAPTER 5**

### **5 Conclusion**

In this fifth and final chapter, the key research findings and contributions of this thesis are summarized, in relation to the research aims and scope defined within the introductory chapter of this thesis. The limitations of the proposed method are also discussed, and future research avenues to address some of these limitations are proposed. Finally, the implications and potential applications of the proposed method for self-organized settlement modeling and generative design as a whole are presented.

#### **5.1 General evaluation**

This thesis aimed to investigate the application of single-agent DRL methods in self-organized settlement modelling, present a framework to formulate any settlement aggregative growth model as an MDP that could be then solved using various DRL algorithms. Previous research already established that the self-organization of settlements is a process where actors or agents perform aggregative sequential actions over time that lead to a continuous state of adaptation within the settlement's form. The MDP framework's characteristics enable the modeling of such local sequential actions within an environment. This thesis proposes the SOS-MDP framework as a guideline to transform any self-organized urban growth model to an MDP. Within this framework, an agent would perform local actions within a geometric generative environment given information about the current state of the settlement and a reward feedback signal that

directs the learning of the agent towards a more favorable settlement growth sequential policy.

The proposed SOS-MDP framework was tested by formulating three environments, the GBR environment with two versions and the GPN environment, based on two cellular automata self-organized urban growth models from the 80s and 90s, namely the ‘Beady Ring’ model (Hillier & Hanson, 1984) and the ‘Path Node Generator’ model (Erickson & Lloyd-Jones, 1997). These three environments were then solved using both the DQN algorithm (Mnih et al., 2013) and the PPO algorithm (Schulman et al., 2017). The experimental results on those three environments indicate that DRL agents could be used to learn settlement self-organized growth that maximize various performance objectives. The PPO agent, in particular, managed to constantly learn improved policies overtime based only on a local observation of the environment, yielding settlement forms that perform gradually better given the preset reward function in each environment. The tested DRL algorithms, however, are not sufficiently sample efficient to converge to optimal policies within a limited training timestep budget and are highly sensitive to hyperparameter choice and model parameter initialization, especially in the case of the DQN algorithm.

The proposed method is a useful contribution in terms of automatically learning self-organized settlement generative policies following a preset feedback signal only, as opposed to having to explicitly formulate the algorithmic rules of aggregation at each timestep to generate or model a particular settlement form using rule-based methods. This method, however, stays limited in terms of its applicability to larger, and more realistic settlement growth models, mainly due to the high computational efficiency in the environment implementation that it requires and the insufficient sample efficiency of DRL algorithms necessitating many millions of timesteps to learn meaningful policies.

## **5.2 Limitations of the method**

The method proposed in this thesis has obvious limitations caused by constraints in scope and computational budget. Some of these limitations and recommendations for future research to address these limitations are as follows:

- Many important performance factors for settlement growth were not included in the experiments in this thesis because most of these metrics are computationally expensive to compute at each timestep. Implementations of environments that include street network performance metrics such as centrality are included in the appendix but were not used due to their polynomial time complexity, rendering training computationally infeasible. Future research should work on **developing computationally efficient environments** that represent more accurately self-organized settlement growth dynamics for RL training.
- DRL methods **require large computational budgets**. This need grows exponentially bigger when the simulation environment gets more complex and closer to reflect real-world settlement growth dynamics. This limits the applicability of the proposed method to real-world scenarios.
- The models used in this thesis **only cover global actions such as adding a whole building or street segment**. The environments could be further developed to expand the action space to include finer grain actions such as room extensions, balcony or staircase additions, etc.
- Single agent methods reach their limits when the observation space grows larger. Multi-agent deep reinforcement learning (MADRL) methods could perform better for settlement growth simulation as each agent would only receive a small observation related to its local domain of action. MADRL is also a better fitting framework to self-organized settlement morphogenesis where multiple actors intervene at a local scale. Future research should **investigate the application of MADRL methods to self-organized settlement modeling**, building upon the current work and expanding it further.
- **Only two DRL algorithms were tested** and trained within a limited computational budget and training timestep budget. These algorithms could be tested more extensively using the same framework and environment implementations used in this thesis' experimental work. Additionally, the **experiments could also be expanded to include other DRL algorithms**.

### 5.3 Potential applications of the method

The RL approach for self-organized urban growth modeling proposed in this thesis offers an advantage over traditional rule-based and other agent-based methods used by previous research efforts. The policy of aggregation in our method is automatically learned based on an objective formulated as a reward feedback signal, this takes out the need to uncover and explicitly program generative rules to model settlement growth using rule-based methods. Previously proposed agent-based methods do not encode the generative decision policy as a function that could be deployed on similar conditions later, the optimization needs to be performed from scratch every time. The proposed learning-based method encodes the policy as learned parameters of a neural network or a function that maps the state space to the action space, therefore it could be used later to simulate settlement growth on other similar conditions without necessitating the retraining of the agent. The proposed method could be used to predict further growth of a settlement given a preset performance objective. It could also be used to uncover sequences of aggregation that would lead to a certain settlement form that satisfies a predefined performance outcome, as well as the optimization of urban form towards an objective or multiple objectives following self-organized bottom-up processes of urban form generation.

These potential applications would of course necessitate the development of more sophisticated simulation environments with more realistic constraints and measures of performance as a prerequisite while maintaining computational efficiency. The applicability of DRL methods to larger simulations of urban growth could also potentially be possible using multi-agent RL setups that would build on top of the current proposed approach.

Additionally, the proposed SOS-MDP framework could be generalized to any performance-based single-agent generative design problem that could be solved through sequential aggregation or a set of sequential design decisions.

## Bibliography

- Akiba, T., Sano, S., Yanase, T., Ohta, T., Koyama, M. (2019). Optuna: A Next-generation Hyperparameter Optimization Framework. In Proceedings of the 25<sup>th</sup> International Conference on Knowledge Discovery and Data Mining.
- Akizuki, Y., Bernhard, M., Kakooee, R., Kladeftira, M., Dillenburger, B. (2020). Generative modeling with design constraints: Reinforcement learning for object generation. RE: Anthropocene, Design in the Age of Humans - Proceedings of the 25th International Conference on Computer-Aided Architectural Design Research in Asia, CAADRIA 2020, 1, 445–454.
- Allen, P.M. (1997). Cities and Regions as Self-Organizing Systems: Models of Complexity. Taylor and Francis, London.
- Allen, P.M., & Sanglier, M. (1981). Urban Evolution, Self-Organization, and Decisionmaking. Environment and Planning A: Economy and Space, 13(2), 167–183. <https://doi.org/10.1068/a130167>
- Apolinarska, AA, Pacher, M, Li, H, Cote, N, Pastrana, R, Gramazio, F and Kohler, M (2021). Robotic assembly of timber joints using reinforcement learning. Automation in Construction, 125, p. 103569.
- Arefi, M. (2011). Order in informal settlements: A case study of Pinar, Istanbul. Built Environment, 37(1), 42–56. <https://doi.org/10.2148/benv.37.1.42>
- Barros, P., Beirão, J., & Duarte, J. P. (2013). The Language of Mozambican Slums: Urban integration tool for Maputo's informal settlements. ECAADe, 31, 715–724.

- Batty M (1971) Modelling Cities as Dynamic Systems. *Nature* 231: 425–428
- Batty, M. (1972). An Experimental Model of Urban Dynamics. *The Town Planning Review*, 43(2), 166–186. <http://www.jstor.org/stable/40102816>
- Batty, M. (2005). *Cities and Complexity: Understanding Cities with Cellular Automata, Agent-Based Models, and Fractals*. Cambridge, MA: The MIT Press.
- Batty, M. (2008). Fifty Years of Urban Modeling: Macro-Statics to Micro-Dynamics. In: Albeverio, S., Andrey, D., Giordano, P., Vancheri, A. (eds) *The Dynamics of Complex Urban Systems*. Physica-Verlag HD. [https://doi.org/10.1007/978-3-7908-1937-3\\_1](https://doi.org/10.1007/978-3-7908-1937-3_1)
- Batty, M., Xie, Y. and Sun, Z. (1999). Modeling Urban Dynamics through GIS-based Cellular Automata. *Computers, Environment and Urban Systems*, 23, 205-233. [http://dx.doi.org/10.1016/S0198-9715\(99\)00015-0](http://dx.doi.org/10.1016/S0198-9715(99)00015-0)
- BuHamdan, S., Alwisy, A., Danel, T., Bouferguene, A., Lafhaj, Z. (2022). The use of reinforced learning to support multidisciplinary design in the AEC industry: Assessing the utilization of Markov Decision Process. *International Journal of Architectural Computing*, 20(2), 216-237.
- Bura, S., Guérin-Pace, F., Mathian, H., Pumain, D., Sanders, L. (1996). Multi-agent systems and the dynamics of a settlement system. *Geographical Analysis*, 28(6), 161–178.
- Brockman, G., Cheung, V., Pettersson, L., Schneider, J., John, S., Jie, T., Zaremba, W. (2016). OpenAI Gym. arXiv:1606.01540v1.
- Çalışkan, O (2017). Parametric Design in Urbanism: A Critical Reflection. *Planning Practice & Research*, 32(4), 417-443.

Cecchini, A., Low, N., & Johnston, R. (2000). Review: Cities and Regions as Self-Organizing Systems: Models of Complexity, Transnational Classes and International Relations, *Becoming a Geographer. Environment and Planning A: Economy and Space*, 32(7), 1325–1330.

<https://doi.org/10.1068/a3207rvw>

Chaillou, S (2020). Archigan: Artificial Intelligence X Architecture. *Architectural Intelligence*, Singapore, 117-127.

Chaillou, S. (2022). *Artificial Intelligence and Architecture: From Research to Practice*. Berlin, Boston: Birkhäuser.

Chang, S., Saha, N., Castro-Lacouture, D., Pei-Ju Yang, P. (2019). Generative design and performance modeling for relationships between urban built forms, sky opening, solar radiation and energy, *Energy Procedia*, 158, 3994-4002.

Chatzipoulka, C., Compagnon, R., Nikolopoulou, M. (2016). Urban geometry and solar availability on façades and ground of real urban forms: using London as a case study. *Solar Energy*, 138, 53-66.

Chen, L., Lu, K., Rajeswaran, A., Lee, K., Grover, A., Laskin, M., Abbeel, P., Srinivas, A., & Mordatch, I. (2021). Decision Transformer: Reinforcement Learning via Sequence Modeling. 1–21.

Chirkin, A. M., & König, R. (2016). Concept of interactive machine learning in urban design problems. *Conference on Human Factors in Computing Systems - Proceedings*, 07-May-201(May 2016), 10–13.

Cho, D., Kim, J., Shin, E., Choi, J., & Lee, J. K. (2020). Recognizing Architectural Objects in Floor-plan Drawings Using Deep-learning Style-transfer Algorithms. *Proceedings of the 25th International Conference on*

Computer-Aided Architectural Design Research in Asia, CAADRIA 2020, 2, 719–727.

Chokhachian, A., Perini, K., Giulini, S., Auer, T. (2020). Urban performance and density: Generative study on interdependencies of urban form and environmental measures. *Sustainable Cities and Society*, 53, 101952.

Collins, C., Dennehy, D., Conboy, K., Mikalef, P. (2021). Artificial intelligence in information systems research: A systematic literature review and research agenda. *International Journal of Information Management*, 60, 102383.

Daffertshofer, A., Haken, H., & Portugali, J. (2001). Self-Organized Settlements. *Environment and Planning B: Planning and Design*, 28(1), 89–102.

del Campo, M. & Manninger, S. (2019). Imaginary Plans. *Proceedings of the 2019 ACADIA Conference – Ubiquity and Autonomy*, Austin, 412-418.

del Campo, M., Carlson, A., & Manninger, S. (2020). Towards Hallucinating Machines - Designing with Computational Vision. *International Journal of Architectural Computing*, 1–16.

Dias, M. A. (2014). Informal Settlements: A Shape Grammar Approach. *Journal of Civil Engineering and Architecture*, 8(11), 1389–1395.

Dovey, K. (2012). Informal urbanism and complex adaptive assemblage. *International Development Planning Review*, 34(4), 349–368.  
<https://doi.org/10.3828/idpr.2012.23>

Dovey, K. (2019). Informal Settlement as a Mode of Production. In *The New Companion to Urban Design* (pp. 139–151).  
<https://doi.org/10.4324/9780203731932-15>

- Dovey, K., & Kamalipour, H. (2018). Informal / Formal Morphologies. In Mapping urbanities (pp. 223–244). Routledge, New York.
- Dovey, K., van Oostrum, M., Chatterjee, I., & Shafique, T. (2020). Towards a morphogenesis of informal settlements. *Habitat International*, 104, 102240. <https://doi.org/10.1016/J.HABITATINT.2020.102240>
- Du, Z., Shen, H., Li, X., Wang, M. (2020). 3D building fabrication with geometry and texture coordination via hybrid GAN. *Journal of Ambient Intelligence and Humanized Computing*, doi:10.1007/s12652-020-02488-9.
- Duarte, J. P., Rocha, J. M., & Soares, G. D. (2007). Unveiling the structure of the Marrakech Medina: A shape grammar and an interpreter for generating urban form. *Artificial Intelligence for Engineering Design, Analysis and Manufacturing: AIEDAM*, 21(4), 317–349.
- Erickson, B. & Lloyd-Jones, T. (1997). Experiments with Settlement Aggregation Models. *Environment and Planning B: Planning and Design*, 24(6), 903–928.
- Forrester, J. W. (1970). Urban Dynamics. *IMR; Industrial Management Review* (Pre-1986), 11(3), 67-69.
- Goodfellow, I, Bengio, Y. and Courville, A. (2016). *Deep Learning*. MIT Press, Cambridge.
- Gouverneur, D. (2015). Planning and Design for Future Informal Settlements Shaping the Self-Constructed City. In *Angewandte Chemie International Edition*, 6(11), 951–952. (1st Edition). Routledge.
- Hakim, B. S. (1986). *Arabic-Islamic Cities: Building and Planning Principles*. (2nd edn., 1988), London.

- Hakim, B. S. (2008). Mediterranean urban and building codes: Origins, content, impact, and lessons. *Urban Design International*, 13(1), 21–40.
- Hakim, B. S. (2014). *Mediterranean Urbanism: Historic Urban / Building Rules and Processes* (1st ed.). Springer.
- Hakim, B.S., & Ahmed, Z. (2006). Rules for the built environment in 19th century Northern Nigeria. *Journal of Architectural and Planning Research*, 23(1), 1–26.
- Han, Z., Yan, W., & Liu, G. (2021). A Performance-Based Urban Block Generative Design Using Deep Reinforcement Learning and Computer Vision. In *Proceedings of the 2020 DigitalFUTURES*. Springer, Singapore, 134-143.
- Harding, J (2016). Dimensionality Reduction for Parametric Design Exploration. *Advances in Architectural Geometry 2016*, Zurich, 274-287.
- Heath, B., Hill. R., Ciarallo, F. (2009). A survey of agent-based modeling practices. *The Journal of Artificial Societies and Social Simulation*, 12(4), 9.
- Henderson, P., Islam, R., Bachman, P., Pineau, J., Precup, D., & Meger, D. (2018). Deep reinforcement learning that matters. *Proceedings of the AAAI Conference on Artificial Intelligence*, 32(1).
- Heppenstall, A., Malleson, N., & Crooks, A. (2016). Space, the Final Frontier: How Good are Agent-Based Models at Simulating Individuals and Space in Cities? *Systems*, 4(1), 9.
- Hillier, B., Greene, M., Desyllas, J. (2000). Self-Generated Neighborhoods: The Role of Urban Form in the Consolidation of Informal Settlements. *Urban Design International*, 5, 61–96.

- Hillier, B., & Hanson, J. (1984). *The Social Logic of Space*. Cambridge, New York: Cambridge University Press.
- Holland, J.H. (1995). *Hidden Order: How Adaptation Builds Complexity*. Helix Books, Addison-Wesley, NY.
- Holland, J.H. (2006). Studying Complex Adaptive Systems. *Journal of Systems Science and Complexity*, 19(1), 1-8.
- Hoshen, J., & Kopelman, R. (1976). Percolation and cluster distribution. I. Cluster multiple labeling technique and critical concentration algorithm. *Physical Review B*, 14(8), 3438-3445. <https://doi.org/10.1103/PhysRevB.14.3438>
- Huang, C. (2021). Reinforcement learning for architectural design-build: Opportunity of Machine Learning in a Material-informed Circular Design Strategy. *Proceedings of the 26th International Conference of the Association for Computer-Aided Architectural Design Research in Asia (CAADRIA) 2021*, 1, 171-180.
- Huang, W., Zheng, H. (2018). Architectural drawings recognition and generation through machine learning. *Recalibration on Imprecision and Infidelity, Proceedings of the 38th Annual Conference of the Association for Computer Aided Design in Architecture, ACADIA 2018*, 156-165.
- Iovene, M., De Córdova, G.F., Romice, O., Porta, S. (2017). Towards Informal Planning: Mapping the Evolution of Spontaneous Settlements in Time. In *Proceedings of the 24th ISUF 2017 - City and Territory in the Globalization Age*, Universitat Politècnica de València, Valencia, Spain.
- Jacobs, J. (1961). *The death and life of great American cities*. Vintage Books.
- Jang, S., Yoo, S., & Kang, N. (2020). *Generative Design by Reinforcement*

Learning: Enhancing the Diversity of Topology Optimization Designs.

Jones, P. (2019). The Shaping of Form and Structure in Informal Settlements: A Case Study of Order and Rules in Lebak Siliwangi, Bandung, Indonesia. *Journal of Regional and City Planning*, 30(1), 43-61.

Joyce, S.C., & Nazim, I. (2019). Limits to Applied ML in Planning and Architecture Understanding and defining extents and capabilities. Towards a New, Configurable Architecture - Proceedings of the 39th eCAADe Conference, 1, 243-252.

Kamalipour, H., Dovey, K. (2020). Incremental Production of Urban Space: A Typology of Informal Design. *Habitat International*, 98, 102133.

Kamalipour, H. (2016). Forms of Informality and Adaptations in Informal Settlements. *International Journal of Architectural Research*, 10, 60-75.

Kamalipour, H., & Iranmanesh, A. (2021). Morphogenesis of Emerging Settlements: Mapping Incremental Urbanism. *MDPI Land* 2021, 10(1), 1-16.

Kelly, T., Guerrero, P., Steed, A., Wonka, P., Mitra, N.J. (2018). FrankenGAN: guided detail synthesis for building mass models using style-synchronized GANs. *ACM Transactions on Graphics*, 37, 1-14.

Koenig, R., & Bauriedel, C. (2009). Generating Settlement Structures: A Method for Urban Planning and Analysis Supported by Cellular Automata. *Environment and Planning B: Planning and Design*, 36(4), 602-624. <https://doi.org/10.1068/b34025>

Koenig, R., Miao, Y., Aichinger, A., Knecht, K., & Konieva, K. (2020). Integrating urban analysis, generative design, and evolutionary optimization for

solving urban design problems. *Environment and Planning B: Urban Analytics and City Science*, 47(6), 997-1013.

Koenig, R. & Varoudis, T.(2016). Spatial Optimizations: Merging depthmapX, spatial graph networks and evolutionary design in Grasshopper.  
<https://nbn-resolving.org/urn:nbn:de:gbv:wim2-20160622-26040>

Kolarevic, B., & Klinger, K. (2008). *Manufacturing Material Effects: Rethinking Design and Making in Architecture*. Routledge.

Leach, N. (2015). (In)formational Cities. *Architectural Design, Mass-Customized Cities*, 85(6), 64-69. <https://doi.org/10.1002/ad.1979>

Liu, R. (2020). Agent-Based Simulation Research on Urban Land Development of San Diego. *Current Urban Studies*, 8, 658-687.

Luis, S.Y., Peralta Samaniego, F., Reina, D.G., Toral Marín, S. (2021). A Sample-Efficiency Comparison Between Evolutionary Algorithms and Deep Reinforcement Learning for Path Planning in an Environmental Patrolling Mission. *2021 IEEE Congress on Evolutionary Computation (CEC)*, 71-78.

Makki, M., Showkatbakhsh, M., Tabony, A. and Weinstock, M. (2018). Evolutionary Algorithms for Generating Urban Morphology: Variations and Multiple Objectives. *International Journal of Architectural Computing*, 17(1), 5–35.

Mandow, L., Pérez-de-la-Cruz, J.L., Rodríguez-Gavilán, A.B., Ruiz-Montiel, M. (2020). Architectural planning with shape grammars and reinforcement learning: Habitability and energy efficiency. *Engineering Applications of Artificial Intelligence*, 96, 103909.

- Mania, H., Guy, A., Recht, B. (2018). Simple random search provides a competitive approach to reinforcement learning. 32nd Conference on Neural Information Processing Systems (NeurIPS 2018), Montréal, Canada.
- McCorduck, P. (2004). *Machines who think: A personal inquiry into the history and prospects of artificial intelligence* (2nd ed.), A. K. Peters.
- Meadows, D. H. (2008). *Thinking in Systems: A Primer*. Chelsea Green Publishing.
- Meadows, D. H. (2001). Dancing with Systems. In P. M. Senge, C. O. Scharmer, J. Jaworski, & B. S. Flowers (Eds.), *Presence: Human Purpose and the Field of the Future* (pp. 259-282). Society for Organizational Learning.
- Miao, Y., Koenig, R., Knecht, K. (2020). The Development of Optimization Methods in Generative Urban Design: A Review. *SimAUD: Symposium on Simulation for Architecture & Urban Design*, October, 247-254.
- Mintrone, A., & Erioli, A. (2021). Training Spaces - Fostering machine sensibility for spatial assemblages through wave function collapse and reinforcement learning. *The 39th eCAADe Conference*, 1, 17-26.
- Mnih, V., Badia, A. P., Mirza, M., Graves, A., Lillicrap, T. P., Harley, T., Silver, D., Kavukcuoglu, K. (2016). Asynchronous Methods for Deep Reinforcement Learning. *ArXiv. ICML 2016*, <http://arxiv.org/abs/1602.01783v2>.
- Mnih, V., Kavukcuoglu, K., Silver, D., Graves, A., Antonoglou, I., Wierstra, D., Riedmiller, M. (2013). Playing Atari with Deep Reinforcement Learning. *ArXiv. NIPS Deep Learning Workshop 2013*, <http://arxiv.org/abs/1312.5602v1>.
- Mnih, V., Kavukcuoglu, K., Silver, D. et al. (2015). Human-level control through deep reinforcement learning. *Nature*, 518, 529-533.

<https://doi.org/10.1038/nature14236>

Mobaraki, A.; Oktay Vehbi, B. A Conceptual Model for Assessing the Relationship between Urban Morphology and Sustainable Urban Form. *Sustainability* 2022, 14, 2884. <https://doi.org/10.3390/su14052884>

Nauata, N., Chang, K.H., Cheng, C.Y., Mori, G., Furukawa, Y. (2020). House-GAN: Relational generative adversarial networks for graph-constrained house layout generation. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, LNCS, 162-177.

Ororbia, M.E. & Warn, G.P. (2022). Design Synthesis Through a Markov Decision Process and Reinforcement Learning Framework. *ASME, Journal of Computing and Information Science in Engineering*, 22(2), 021002.

Orum, A.M. & Chiaradia, A.J.F. (2022). Urban Morphology/Urban Form. In *The Wiley Blackwell Encyclopedia of Urban and Regional Studies*, A.M. Orum (Ed.).

Perlin, K. (2001). Noise hardware. *Real-Time Shading SIGGRAPH Course Notes*, M. Olano (Ed.).

Porta, S., Romice, O., Maxwell, J.A., Russell, P., Baird, D. (2014). Alterations in Scale: Patterns of Change in Main Street Networks across Time and Space. *Urban Stud.*, 51, 3383-3400.

Polidoro, M., Lollo, J., and Barros, M. (2012). Urban Sprawl and the Challenges for Urban Planning. *Journal of Environmental Protection*, 3(9), 1010-1019. <https://doi.org/10.4236/jep.2012.39117>.

Preisinger, C. (2013). Linking Structure and Parametric Geometry. *Architectural*

Design, 83, 110-113

- Qeisi, S. Al, & Al-Alwan, H. (2021). Generative Urban Design Concepts and Methods: A Research Review. IOP Conference Series: Materials Science and Engineering, 1090(1), 012085.
- Quan, S. J., Park, J., Economou, A., & Lee, S. (2019). Artificial intelligence-aided design: Smart Design for sustainable city development. Environment and Planning B: Urban Analytics and City Science, 46(8), 1581-1599.
- Raffin, A., Hill, A., Gleave, A., Kanervisto, A., Ernestus, M., Dormann, N. (2021). Stable-Baselines3: Reliable Reinforcement Learning Implementations. Journal of Machine Learning Research, 22, 1-8.
- Rahimian, M., Beirão, J.N., Duarte, J.P., Iulo, L.D. (2020). A Grammar-Based Generative Urban Design Tool Considering Topographic Constraints: The Case for American Urban Planning. ECAADe 37 / SIGRaDi 23 - Data - City Information Modelling and GIS, September, 267-276.
- Rapoport, A. (1977). Human Aspects of Urban Form: Towards a Man-Environment Approach to Urban Form and Design, Urban and Regional Planning Series 15. Oxford: Pergamon Publishing.
- Romice, O., Thwaites, K., Porta, S., Greaves, M., Barbour, G., Pasino, P. (2017). Urban Design and Quality of Life. In: Fleury-Bahi, G., Pol, E., Navarro, O. (eds) Handbook of Environmental Psychology and Quality of Life Research. International Handbooks of Quality-of-Life. Springer, Cham. [https://doi.org/10.1007/978-3-319-31416-7\\_14](https://doi.org/10.1007/978-3-319-31416-7_14)
- Rutten, D. (2013). Galapagos: On the Logic and Limitations of Generic Solvers. Architectural Design, 83(2), 132-135.

- Schulman, J., Wolski, F., Dhariwal, P., Radford, A. & Klimov, O. (2017). Proximal Policy Optimization Algorithms. ArXiv, <http://arxiv.org/abs/1707.06347>.
- Schumacher, P. (2011). Parametricism: A New Global Style for Architecture and Urban Design. *AD Architectural Design*, 81(4), 14-23. doi: 10.1002/ad.1274
- Shafique, T. I. (2021). Desiring Karail: Morphogenesis of an informal settlement in Dhaka (PhD thesis).
- Shi, X & Yang, W (2013). Performance-driven architectural design and optimization technique from the perspective of architects. *Automation in Construction*, 32, 125-135.
- Silver, D. (2015). Lectures on Reinforcement Learning. UCL.
- Sims, D. (2010). Understanding Cairo: The Logic of a City Out of Control. New York: American University in Cairo Press.
- Song, Y., & Yuan, P.F. (2021). A Research on Building Cluster Morphology Formation Based on Wind Environmental Performance and Deep Reinforcement Learning. *eCAADe* 39, 1, 335-344.
- Sun, H., & Ma, L. (2020). Generative design by using exploration approaches of reinforcement learning in density-based structural topology optimization. *Designs*, 4(2), 1-20.
- Sutton, R.S., & Barto, A.G. (1998). Reinforcement Learning: An Introduction. The MIT Press, Cambridge.
- Szepesvári, C. (2009). Algorithms for reinforcement learning. In *Synthesis Lectures on Artificial Intelligence and Machine Learning*.

- Turing, A.M. (1950). Computing Machinery and Intelligence. *Mind*, 59(236), 433-460.
- Uzun, C., Colakoglu, M., Inceoglu, A. (2020). GAN as a generative architectural plan layout tool: A case study for training DCGAN with Palladian plans and evaluation of DCGAN outputs. *ITU Journal of the Faculty of Architecture* 17, 185-198.
- van Damme, E. (2001). Game Theory: Noncooperative Games. *International Encyclopedia of the Social & Behavioral Sciences*, Pergamon, 5873-5880.
- Veloso, P., & Krishnamurti, R. (2020). An Academy of Spatial Agents: Generating spatial configurations with deep reinforcement learning. *eCAADe38*, 2, 191-200.
- Verniz, D., & Duarte, J. P. (2017). Santa Marta Urban Grammar: Towards an understanding of the genesis of form. *35th ECAADe*, Rome, Italy, 2(4), 477-484.
- Vierlinger, R. (2013). Multi Objective Design Interface. Master's Thesis, Technischen Universität Wien, Fakultät für Bauingenieurwesen, Wien, Austria.
- Wang, D., Snooks, R. (2021). Artificial Intuitions of Generative Design: An Approach Based on Reinforcement Learning. In: Yuan, P.F., Yao, J., Yan, C., Wang, X., Leach, N. (eds) *Proceedings of the 2020 DigitalFUTURES. The International Conference on Computational Design and Robotic Fabrication (CDRF) 2020*. Springer, Singapore, 189-198.
- Wang, D., & Snooks, R. (2021). Intuitive behavior: The operation of reinforcement learning in generative design processes. *Proceedings of the 26th International Conference of the Association for Computer-Aided*

Architectural Design Research in Asia, CAADRIA 2021, 1, 101-110.

Wang, D., Fu, Y., Wang, P., Huang, B., Lu, C.T. (2020). Reimagining city configuration: Automated urban planning via adversarial learning. Proceedings of the ACM International Symposium on Advances in Geographic Information Systems, Association for Computing Machinery, 497-506.

Wibranek, B., Liu, Y., Funk, N., Belousov, B., Peters, J., & Tessmann, O. (2021). Reinforcement Learning for Sequential Assembly of SLBlocks: Self-Interlocking Combinatorial Design Based on Machine Learning. eCAADe 2021: Towards a New, Configurable Architecture, 1, 27-36.

Wortmann, T & Nannicini, G (2017). Introduction to architectural design optimization. In Karakitsiou, A, Migdalas<sup>2</sup>, A, Rassia, ST and Pardalos<sup>4</sup>, PM (eds) 2017, City Networks, Springer, 259-278.

Zhang, H., Blasetti, E. (2020). 3D architectural form style transfer through machine learning. Proceedings of the 25th International Conference on Computer-Aided Architectural Design Research in Asia, CAADRIA 2020, The Association for Computer-Aided Architectural Design Research in Asia (CAADRIA), 661-670.

Zhang, Y., Vuong, Q. H., Song, K., Gong, X. Y., & Ross, K. W. (2018). Efficient entropy for policy gradient with multi-dimensional action space. 6th International Conference on Learning Representations, ICLR 2018 - Workshop Track Proceedings.

Zheng, H., & Yuan, P. F. (2021). A generative architectural and urban design method through artificial neural networks. Building and Environment, 205, 108178

## Appendix

This appendix includes further details on the setup of the RL experiments included in this thesis, as well as code contributions to ensure the reproducibility of the results of this thesis as much as possible and support future research into applications of RL in urban growth modeling and generative design as a whole.

### Experiment Setup Details

The implementation of the environments developed for experimentation in this thesis, as well as RL agent training and hyperparameter tuning is publicly shared to encourage both RL and architecture and urban design communities to use it for reproducing the results, testing various RL algorithms, and building further on top of this research. The code is available in the following GitHub repository [https://github.com/houssameehsain/SOS\\_MDP](https://github.com/houssameehsain/SOS_MDP).

All the RL environments used for this thesis experimentation are compatible with the OpenAI Gym convention which is widely used within the RL community. This will ease the use of the implementation of the Beady Ring and Path Node environments by future research. The implementations could also be easily extended to incorporate more performance metrics and urban growth dynamics. The OpenAI Gym structure allows for training agents in these environments using various RL algorithms.

This thesis makes use of Stable-Baselines3 (Raffin et al., 2021) implementations of the various RL algorithms (DQN, PPO) to ensure stable training results as well as the reproducibility of those results by others. The hyperparameter tuning process is performed using the Optuna (Akiba et al., 2019) implementation of Bayesian search over a predefined hyperparameter space. Hyperparameter tuning is the most computationally intensive part of training an RL agent. This tuning process was performed for 3 days for each environment and RL algorithm with 300 thousand timesteps per trial parallelized on 32 CPU cores using Google Vertex AI cloud computing service. The training process was also performed using the same cloud high performance computing (HPC) setup.

## Hyperparameters

The following two tables list the best hyperparameters found during the tuning process for each environment that were used for training both the DQN and PPO agents. All the hyperparameter tuning results and the code to run this process on each of the environments using the DQN, PPO, and A2C algorithms are in the following GitHub repository [https://github.com/houssameehsain/SOS\\_MDP/tree/main/hyperparameter\\_tuning](https://github.com/houssameehsain/SOS_MDP/tree/main/hyperparameter_tuning).

| DQN hyperparameters    | GBR v1 (CNN) | GBR v1 (Moore) | GBR v2   | GPB      |
|------------------------|--------------|----------------|----------|----------|
| Batch size             | 256          | 128            | 256      | 64       |
| Buffer size            | 10k          | 50k            | 50k      | 100k     |
| Exploration final rate | 3.76e-02     | 1.04e-03       | 4.48e-03 | 1.12e-02 |
| Exploration fraction   | 1.3e-01      | 3.9e-02        | 6.91e-03 | 3.36e-03 |
| Gamma                  | 0.999        | 0.999          | 0.99     | 0.99     |
| Learning rate          | 6.21e-04     | 8.09e-04       | 1.02e-04 | 3.99e-02 |
| Learning starts        | 5000         | 0              | 5000     | 10k      |
| Number of layers       | 3            | 1              | 1        | 2        |
| Layer size             | 256          | 64             | 64       | 64       |
| Gradient steps         | 256          | 4              | 128      | 16       |
| Target update interval | 5000         | 1              | 20k      | 15k      |

Table a.1 Hyperparameters used to train DQN agents in each environment. Moore refers to the local observation version of the GBR v1, while CNN means Convolutional neural network referring to the global observation version of that same environment. Produced by author.

| PPO hyperparameters | GBR v1   | GBR v2   | GPB      |
|---------------------|----------|----------|----------|
| Activation function | ReLU     | Tanh     | Tanh     |
| Batch size          | 256      | 64       | 64       |
| Clip range          | 0.3      | 0.4      | 0.4      |
| Entropy coefficient | 2.52e-07 | 4.03e-08 | 2.99e-07 |
| GAE lambda          | 0.99     | 0.98     | 0.92     |

| <b>PPO hyperparameters</b>        | <b>GBR v1</b> | <b>GBR v2</b> | <b>GPN</b> |
|-----------------------------------|---------------|---------------|------------|
| <b>Gamma</b>                      | 0.99          | 0.995         | 0.999      |
| <b>Learning rate</b>              | 1.26e-05      | 1.21e-04      | 1.04e-04   |
| <b>Max. gradient norm</b>         | 0.9           | 0.9           | 0.3        |
| <b>Number of epochs</b>           | 20            | 10            | 20         |
| <b>Number of steps</b>            | 2048          | 512           | 256        |
| <b>Number of layers</b>           | 2             | 3             | 2          |
| <b>Layer size</b>                 | 256           | 256           | 256        |
| <b>Orthogonal Initialization</b>  | Yes           | No            | Yes        |
| <b>Value function coefficient</b> | 2.69e-06      | 4.97e-04      | 9.5e-04    |

Table a.2 Hyperparameters used to train PPO agents in each environment. Produced by author.

### Perlin Simplex Noise Parameters

To reproduce the topographic conditions (introvert, linear, extrovert) used within the GPN environment, the parameters to generate those conditions are as follows:

| <b>Perlin noise parameters</b> | <b>Introvert</b> | <b>Linear</b> | <b>Extrovert</b> |
|--------------------------------|------------------|---------------|------------------|
| <b>Starting location</b>       | 50, 50           | 90, 90        | 50, 40           |
| <b>Scale</b>                   | 30               | 40            | 50               |
| <b>Octaves</b>                 | 3                | 2             | 2                |
| <b>Persistence</b>             | 0.4              | 0.4           | 0.4              |
| <b>Lacunarity</b>              | 2.0              | 2.0           | 1.0              |
| <b>Central flatness</b>        | 0.5              | 0.5           | 0.5              |
| <b>Frequency</b>               | 1                | 1             | 1                |
| <b>Amplitude</b>               | 1                | 1             | 1                |
| <b>Initial value</b>           | 0                | 0             | 1                |
| <b>Random seed</b>             | 77               | 95            | 123              |

Table a.3 Perlin Simplex noise parameters used to generate the topographic conditions within the GPN environment.

Produced by author.

## Markov Chains and the Markov Property

This section defines the Markov property and provides a formal mathematical formulation of it.

The state, in RL literature, is whatever is the current context within the environment at a specific time. The observation, on the other hand, is whatever information the agent receives at each step as a representation of the state of the environment. This observation could be a complete view of the state, but it could also be a partial representation of that state. For example, within the game of chess, each player can view the whole board at each playing turn, therefore they have a full observation of the environment. Within the game of poker, however, the players only have information about their own cards and cannot see other players' cards or the ones in the hands of the dealer. In the case of poker, the observation is only partial. In any case, what is important in terms of observation of the environment's state is not whether it is a full or partial representation, but whether it is a representation that is sufficient to be able to choose an action given the current state in the environment that maximizes reward. Thus, the observation must include immediate information about what is happening in the environment as well as information about past events within that environment. The observation emanating from the environment does not have to inform the agent of everything within that environment. Nevertheless, it must be a compact version of all the relevant information about events that occurred within that environment that would enable the agent to take the appropriate action at the next timestep. An observation signal that retains all this relevant information for decision-making at each step is said to be Markov or hold the Markov property (Sutton & Barto, 1998). For instance, a full knowledge of where chess pieces lie on the board at any given position would serve as a Markov observation. This observation is all a chess player would need to make the next move. Much of the information about previous moves that led to the current position is lost, however, full observation of the board is all that is needed at each step. It summarizes and retains all the necessary important information that would inform the next move. The current observation in a chess environment is Markov because it is totally independent from the sequence of moves and changes to the environment that

happened earlier. One can start at any given chess position and be able to reason about the current state in the game and choose what is thought to be the best move. An environment with Markov observations at each step is also referred to as a Markov environment.

If the environment did not have the Markov property, the next response by the environment would have to depend on all the actions and transitions that happened earlier within the environment. The dynamics of the environment in that case, for a finite set of possible states that the environment could be in, would be described through the following probability distribution:

$$\Pr\{R_{t+1} = r, S_{t+1} = s' \mid S_0, A_0, R_1, \dots, S_{t-1}, A_{t-1}, R_t, S_t, A_t\} \quad a.1$$

for all possible rewards  $r$ , and all possible states  $s'$ , and all possible values of the past events:  $S_0, A_0, R_1, \dots, S_{t-1}, A_{t-1}, R_t, S_t, A_t$  (Sutton & Barto, 1998). The probability distribution here gives the probability of having the next response by the environment in the form of the next state  $S_{t+1}$  and the reward  $R_{t+1}$  given all the past events that occurred within the environment. If the environment has the Markov property, then the environment's response at the next step would only depend on the action and observation of the immediate previous step. The transition dynamics within that environment could then be described by the following formula:

$$p(s', r \mid s, a) = \Pr\{R_{t+1} = r, S_{t+1} = s' \mid S_t, A_t\} \quad a.2$$

for all possible next states  $s'$ , a reward signal  $r$ , and previous timestep state  $S_t$  and action  $A_t$  (Sutton & Barto, 1998). In this case, the observation of state  $s$  retains all the information necessary to take action  $a$  that then results in reward  $r$  and a transition in the environment's state to state  $s'$  with a probability  $p$ . The environment is Markov if and only if the probability distribution a.2 is equal to the one in a.1 for all possible states and

rewards and all possible sequences of transition within that environment (Sutton & Barto, 1998). If the environment has the Markov property, then following equation a.2 it is possible to predict the next state and the expected reward in the future given the current state the environment is in and the action chosen by the agent. This characteristic makes it therefore possible to estimate the value of each state and the value of each action given a certain state. Therefore, the agent can optimize its policy of action iteratively to maximize the yielded reward based only on the information returned by the environment at each single step.

The Markov property is therefore crucial to RL, and essential to the formulation and setup of the environment. The observation signal must be informative so that the agent can learn any meaningful policy. RL methods and algorithms are designed to take advantage of this Markov property with the aim of maximizing rewards. However, they could also be useful in cases where the environment is not exactly Markov, which is the case in a lot of environments that model real-world dynamics (Sutton & Barto, 1998). It is however important to understand and keep in mind the Markov property while designing any RL environment.

## Additional Environments

Additional environments with street networks represented as mathematical graphs are also available in the same GitHub repository. These are extensions of both GBR and GPN environments to include street network performance metrics in the reward signal the agent receives (Figure a.1). These environments were not included in the results of this thesis however due to the intractability of the computation of tuning and training within these environments. The algorithms to compute network centrality, for instance, at each timestep have polynomial time complexity and therefore the computation becomes slower as the settlement grows. It was not thus possible to perform hyperparameter tuning and training on these environments within the computation budget constraint. Future research could use these implementations and build upon them to include street network performance considerations to the modeling of settlements using RL.

Episode 12 Step 205 Return 0.92



*Figure a.1 Sample settlement representation output within the GBR v3 environment. The street network is represented in blue segments. Produced by author.*

The environments are also implemented within Rhinoceros 3D and Grasshopper software (Figure a.2). These additional implementations are available in the following GitHub repository [https://github.com/houssameehsain/SOS\\_MDP/tree/main/RhGh\\_envs](https://github.com/houssameehsain/SOS_MDP/tree/main/RhGh_envs). These implementations were not used due to limitations in connecting and synchronizing the Grasshopper simulation environment with a python-based implementation of the different RL algorithms. Additionally, since Google cloud based HPC resources were used to perform tuning and training of these RL agents, the environments had to be implemented entirely in the Python programming language.

The RL training loop was implemented in connection between the Grasshopper simulation environment and the RL algorithms written in Python using three distinct methods:

- HoopSnake component in Grasshopper with a WebSocket connection to the Python-based RL algorithm (Figure a.3). This implementation offers full access to Rhino/Grasshopper functionality and simulation capabilities. However, it is heavy and not suitable for runs that last for long periods of time (days) since both Rhino and Grasshopper interfaces remain active during computation.
- GhPython component While loop command in Grasshopper with a WebSocket connection to the Python-based RL algorithm (Figure a.4).

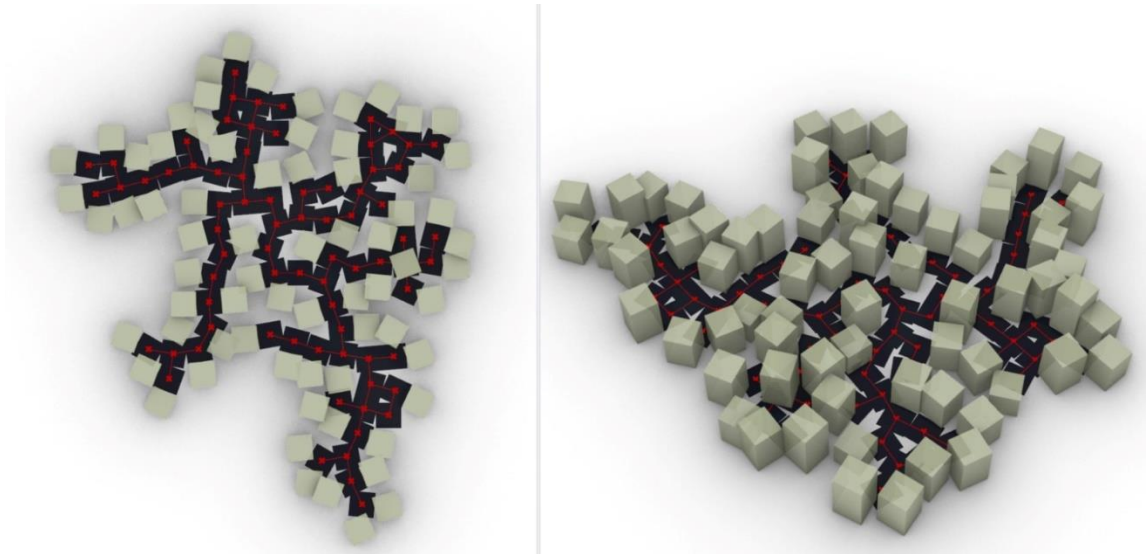


*Figure a.2 Preview of settlement generated within an environment implemented in Rhino/Grasshopper. Produced by author.*

- 

The screenshot shows the Grasshopper Python Script Editor with the following code:

```

File Edit Tools Mode Help
Test OK

16 import ghythonlib.treehelpers as th
17
18
19 class Connection:
20     def __init__(self, s):
21         self._socket = s
22         self._buffer = ""
23
24     def receive_object(self):
25         while len(self._buffer) < 4 or len(self._buffer) < struct.unpack("cL",
26             new_bytes = self._socket.recv(16)
27             if len(new_bytes) == 0:
28                 return None
29             self._buffer += new_bytes
30             length = struct.unpack("cL", self._buffer[4:])[0]
31             header, body = self._buffer[4:], self._buffer[4:length + 4]
32             obj = pickle.loads(body)
33             self._buffer = self._buffer[length + 4:]
34             return obj
35
36     def send_object(self, d):
37         body = pickle.dumps(d)
38         header = struct.pack("cL", len(body))
39         msg = header + body
40         self._socket.send(msg)
41
42 class BeadRingEnv:
43     def __init__(self):
44         self._carrier_color = 127.5
45         self._house_color = 0
46         self._street_color = 255
47         self._cell_size = 3
48         self._env_size = 36

```

The script is connected to a Grasshopper loop component, which is part of a larger network diagram. The loop component is labeled "Loop" and "False". The output of the loop component is connected to a "loop" component, which is labeled "loop" and "out". The "loop" component has a "217ms" delay.

101