A Performance-Based Urban Block Generative Design Using Deep Reinforcement Learning and Computer Vision

Zhen Han[1], Wei Yan[2], and Gang Liu[1](✉)

[1] Tianjin Key Laboratory of Architectural Physical Environment and Ecological Technologies,
Tianjin University, Tianjin, China
dkhertz@163.com, lglgmike@163.com

[2] Department of Architecture, College of Architecture, Texas A&M University, College Station,
TX, USA
wyan@tamu.edu

Abstract. In recent years, generative design methods are widely used to guide urban or architectural design. Some performance-based generative design methods also combine simulation and optimization algorithms to obtain optimal solutions. In this paper, a performance-based automatic generative design method was proposed to incorporate deep reinforcement learning (DRL) and computer vision for urban planning through a case study to generate an urban block based on its direct sunlight hours, solar heat gains as well as the aesthetics of the layout. The method was tested on the redesign of an old industrial district located in Shenyang, Liaoning Province, China. A DRL agent - deep deterministic policy gradient (DDPG) agent - was trained to guide the generation of the schemes. The agent arranges one building in the site at one time in a training episode according to the observation. Rhino/Grasshopper and a computer vision algorithm, Hough Transform, were used to evaluate the performance and aesthetics, respectively. After about 150 h of training, the proposed method generated 2179 satisfactory design solutions. Episode 1936 which had the highest reward has been chosen as the final solution after manual adjustment. The test results have proven that the method is a potentially effective way for assisting urban design.

Keywords: Generative design · Deep reinforcement learning · Urban design · Performance · Hough transform

1 Introduction

Generative design was proposed first in the 1970s and was used in architectural design in 1974 (Frazer 2002). Since then, many research projects utilized different approaches, like cellular automata (CA) and shape grammar (SG), to help designers with their designs. Generative design methods are developed in order to automatically create new design schemes based on the rules or constraints set by designers. In some cases, the performance evaluation is embedded into the generative design methods to drive the creation of

P. F. Yuan et al. (Eds.): CDRF 2020, *Proceedings of the 2020 DigitalFUTURES*, pp. 134–143, 2021.
https://doi.org/10.1007/978-981-33-4400-6_13

schemes. Designers will choose an optimal solution from a large number of generated design alternatives.

Lambe and Dongre (2019) proposed a SG method to create an architectural design scheme based on the style of the existing architecture. Contextualism was used in their work to represent the relationship between new designs and the existing surroundings. Ozdemir and Ozdemir (2018) proposed a novel generation method with multi-criteria decision making (MCDM) techniques to generate alternatives for specific architectural models. Li et al. (2018) introduced the concept of circulation into shape grammar. Circulation is a design method used in architectural design that is formed by connecting the points left by indoor or outdoor space movements of human. In that research, the proposed method was tested on a commercial building, and different alternatives of circulation were generated successfully. Eilouti (2019) introduced a reverse engineering technique into generative design method and proposed a parsing tool to decode the morphogenesis in architecture. The method is synthetic, predictive, and generative. Lee et al. (2018) developed a generic Justified Plan Graph (g-JPG) grammar and proposed a hybrid method that combined Space Syntax and shape grammar to find out both the syntactical and grammatical genotypes of designs.

In addition to aesthetics, performance should also be considered in urban and architectural designs. For example, an appropriate design with superior performance will reduce energy consumption and improve human comfort as well. Many researchers have already combined the generative design methods with stochastic optimization algorithms like genetic algorithm (GA) and particle swarm optimization (PSO). Rodrigues et al. (2019) proposed a methodology of performance-based automated architectural design. This method took into consideration urban geometric and energy consumption and it can be used at the early design stages to explore a concept model. Chang et al. (2019) established some building prototypes and used deep reinforcement learning (DRL) to control the arrangement of buildings. All the schemes created by the DRL algorithm were then evaluated by their performance criteria such as energy consumption, sky openings and solar radiation. The best performance scheme under multi-constraints was chosen as the final design. Youssef et al. (2018) proposed a new method on generating the shape of building integrated photovoltaics (BIPV). This method adjusts the shapes or envelopes of the input buildings in order to generate a series of better BIPV shape alternatives. The optimal placement of BIPV for the optimized building is determined. Yavuz et al. (2018) proposed a novel shape grammar to guide the generation of acoustic panels in order to create an optimal indoor acoustic environment, from the generation of 2D geometric to the evolution of 3D acoustic panels. Rodrigues et al. (2018) proposed a step-by-step method to generate and evaluate schemes. An evolutionary program for the Space Allocation Problem (EPSAP) algorithm was introduced into the step-by-step method to create buildings, with an optimization algorithm used to find the optimal solutions. Sun and Rao (2020) proposed a performance-based generative design framework. The Grasshopper plugins Penguin, Butterfly, and Octopus were used to generate schemes, evaluate the performance, and optimize the designs, respectively.

With the help of the simulation and optimization tools, generative design methods can create the alternatives of high performance. Such an automatic generation algorithm is more effective and time-saving than a manual design approach. However, due to the restrictions of the optimization algorithms and rule-based grammars, the performance-based generative design approaches still have room for improvement, for example:

- The number of alternatives is limited in the rule-based generative design approach. The conventional approaches create the schemes according to rules or laws previously set, and that will influence the diversity of the alternatives.
- The number of design variables must be fixed during the optimization process. For example, the length of the genes in GA and the dimensions of the search space in PSO are constant till the algorithm meets the stop criterion. This means the designers need to determine the design variables at the beginning. However, some variables like the number of the buildings (in an urban design case) or the number of the lamps (in an indoor lighting design case) are very difficult to be determined in the beginning of the optimization algorithm.

In this study, a novel generative design approach using deep reinforcement learning and computer vision was proposed. A DRL agent, the deep deterministic policy gradient (DDPG) agent, is used to observe the site and generate a scheme with high-performance.

2 Methodology

Reinforcement learning is a branch of machine learning where an agent learns to handle an unknown environment based on rewards. DRL is a combination of traditional reinforcement learning and deep learning. Compared with the conventional rule-based generative design method, like CA and GA, methods using DRL can train agents to observe the environment and generate an action by themselves. During the training process, the agent will optimize the parameters to take better actions according to the rewards. Without human rules or laws, the agent can conduct a trial-and-error process automatically. Its end-to-end training doesn't need to determine design variables in the optimization process. In other words, this approach only needs the initial condition (the site information) and the goals (1. to generate an urban block which has a certain total building area; and 2. to calculate building performance by simulation tools like Honeybee in Rhino, as accurate as possible). There is no need to provide the algorithm with other information such as the number of buildings or the shape grammar rules to guide the generative process.

2.1 DRL Based Generative Design Framework

The DRL agent contains two parts: a policy and an algorithm. As shown in Fig. 1, a DRL based generative design framework was established using co-simulation with MATLAB and Rhino/Grasshopper, which includes three steps:

STEP 1: At time *t* of an episode, the agent observes the environment (*Observation,* S_t) and the policy takes an optimal action (*Action,* a_t) according to the observation.
STEP 2: According to the action from the agent, the environment will evaluate how successful the action is to achieve the task goal and send a reward (*Reward,* r_t) back

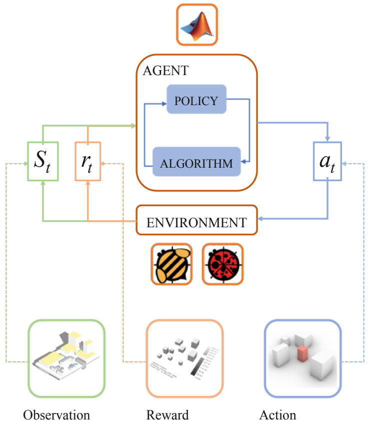


Fig. 1. The framework of the DRL based generative design approach

to the agent. At the same time, the environment will also update its state and send the observation back to the agent.
STEP 3: The algorithm will update the parameters of the policy based on the action a_t, observation S_t and reward r_t. The agent will generate a new action a_{t+1} according to the updated environment S_{t+1}.

The above three steps will repeat in each episode until S_t is a terminal observation. The training process will stop until the maximum episode iterations is reached or the other terminal criterions are met.

2.2 DDPG Agent

The goal of the DRL is to train an agent to take optimal actions to deal with changing of an unknown environment. In this research, the agent was trained using the DDPG algorithm, which is an off-policy, model-free and online DRL approach. The agent will calculate an optimal policy to maximize the long-term reward using actors and critics.

The actor and critic are function approximators used to evaluate the policy and value function. The DDPG agent includes the following four function approximators: an Actor $\mu(S)$; a Target Actor $\mu'(S)$, a Critic $Q(S, a)$ and a Target Critic $Q'(S, a)$. $\mu(S)$ accepts the observation S_t and outputs the optimal action a_t that maximizes the long-term reward; $Q(S, a)$ accepts the observation S_t and action a_t and outputs the prediction of the long-term reward. Both $\mu(S)$ and $\mu'(S)$ and $Q(S, a)$ and $Q'(S, a)$ have the same structure and parameterization. To improve the stability of the DDPG algorithm, $\mu'(S)$ and $Q'(S, a)$ will be updated periodically according to the newest $\mu(S)$ and $Q(S, a)$ parameter values, respectively (Lillicrap et al. 2015). In this research, the $\mu(S)$ and $Q(S, a)$ were established by two deep neural networks based on the observation and action (shown in Figs. 2 and 3, respectively).

As shown in Fig. 2, the actor only receives observation as input, which includes a Site Path and an Index Path. (The details of observation, action and reward is explained

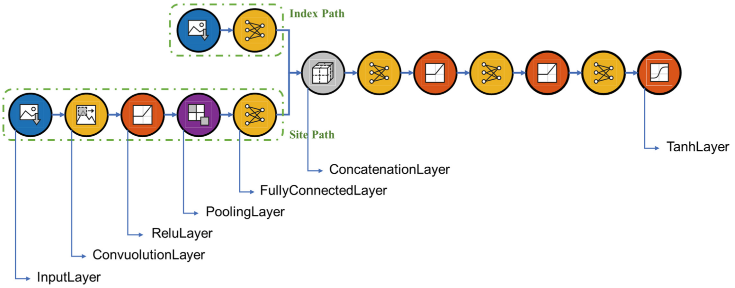


Fig. 2. Actor network structure (utilizing icons from MATLAB)

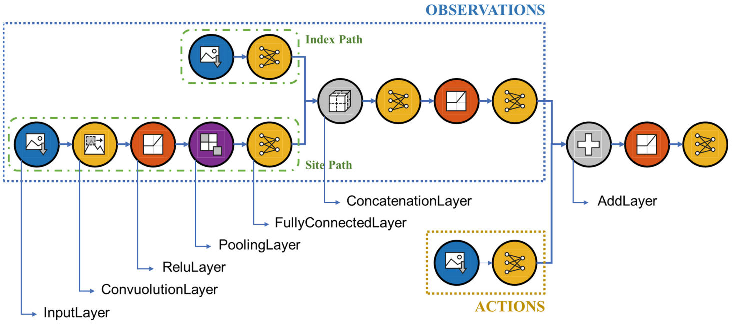


Fig. 3. Critic network structure (utilizing icons from MATLAB)

in Sect. 3.1). The inputs of the Site Path and Index Path are an image and a vector, respectively. A convolution neural network (CNN) is used in the Site Path. As shown in Fig. 3, the critic receives observation and action as well.

2.3 Hough Transform

In computer vision, Hough transform is used to detected lines or curves in an image (Duda and Hart 1972). The Hough transform algorithm can represent a line in the Cartesian space as a point in the Hough space. As shown in Fig. 4, lines in Cartesian space which go through the same point can be described as a curve in Hough space. So, the points on the same line (like Point A, B and C) in Cartesian space must intersect at one point in Hough space. The line in Cartesian space can be described as $r = x\cos\theta + y\sin\theta$, and the coordinate of the intersection point should be (r_0, θ_0) in Hough space.

Gap Distance (GD) is used to describe least distance between two line segments associated with the same Hough transform line. When the distance between the line segments is less than GD, the algorithm will merge the line segments into a single line segment. As shown in Fig. 5(b), five line segments (two blue and three orange) of the detected line segments were used as an example. The Hough transform algorithm will merge them into two line segments when GD was specified as infinity (Fig. 5(c)).

Considering the aesthetics of urban design, this research used Hough transform to evaluate an urban geometric design objective to make sure as many buildings aligned

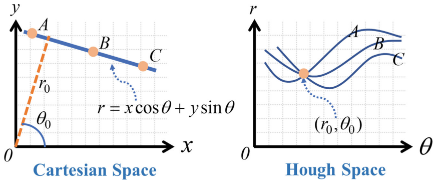


Fig. 4. The principle of Hough transform

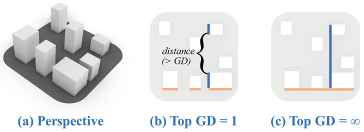


Fig. 5. The diagram of Gap Distance

as possible (as an example objective). Thus, after making GD to infinity, all the line segments in a same line will be merged to one. The fewer lines found after Hough transform means the more buildings are aligned with each others.

3 Case Study

3.1 Observation, Action and Reward

The observation in this research consists a 150 pixel-by-150 pixel-by-3 channel image representing solar radiation performance and a 3-by-1 vector representing building configurations. As shown in Fig. 6(a), the direct sunlight hours nephogram of the site calculated by a Grasshopper plugin, Honeybee, will be resized to 150 × 150 × 3 and sent to the DDPG agent as one part of the observation. Another part of the observation is a vector which consists three elements: total building area, building coverage and floor area ratio (FAR), respectively.

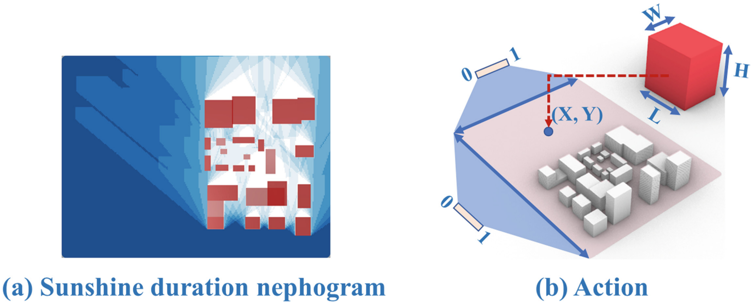


Fig. 6. The diagram of observation, action and reward

In one episode, the agent will arrange one building at one time until the episode is terminated. The action in this research is a 5-by-1 vector consisting of building location

X, location Y, length L, width W and height H. As shown in Fig. 6(b), the location X and location Y are two parameters normalized to a range of 0–1.

The reward function described in Formula (1) consists of six terms: (1) a solar heat gains reward R_{SHG} which is the average solar heat gain of the buildings in winter (*kW/h*); (2) a direct sunlight hours reward R_{SD} which is the average direct sunlight hours of the block on winter solstice (*h*); (3) an aesthetics reward $R_a = 4n - N$ where n is the number of buildings; N is the number of lines determined by Hough transform; (4) a constant reward $R_c = 10$ which encourages the agent to avoid termination; (5) a collision punishment $R_{cp} = -0.5$ and (6) a collision termination punishment $R_{ctp} = -30$.

$$R = \sqrt{R_{SHG}} + 1.5R_{SD} + 5R_a + R_c + R_{cp} + R_{ctp} \tag{1}$$

The coefficients and constants in Formula (1) were determined by a significant volume of tests that can make the agent performs best and they are used to make sure each item have the same order of magnitude (range between 0 to 30).

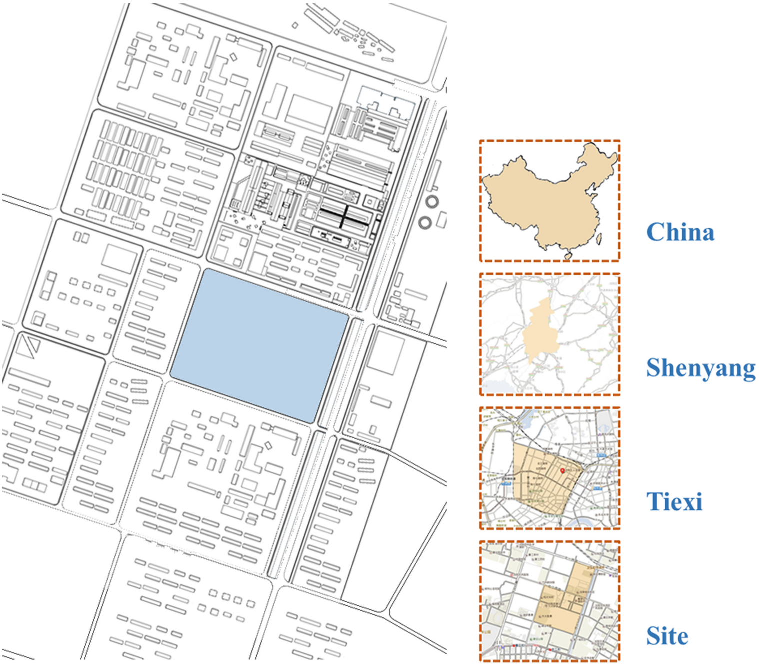


Fig. 7. Site information

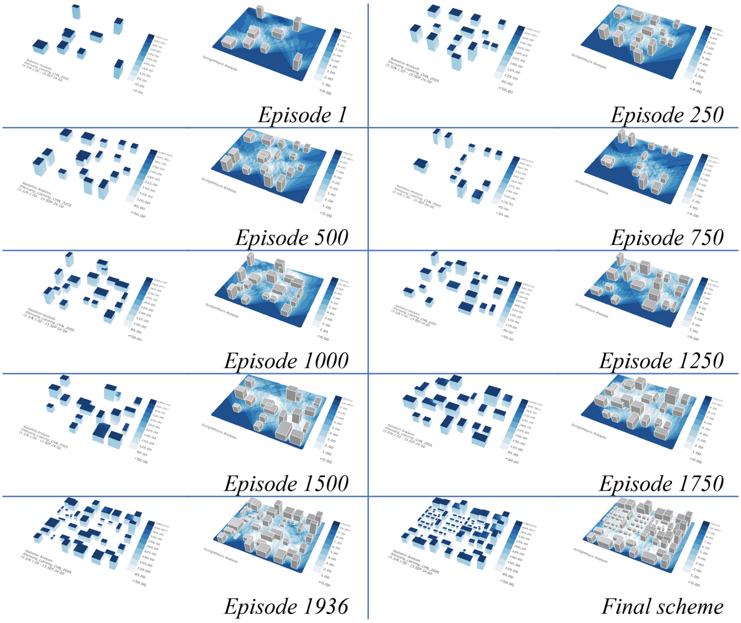
In each episode, the agent will generate the urban block step by step. One building will be created at each step according to the environment until the agent meet the following terminal criterion (1) overlap of two buildings exceeds 50%; and (2) the FAR is over 3. And the environment will be reset to start a new episode until the training process is over.

3.2 Site Information

With the acceleration of urbanization in China, the old industrial districts in cities are being rebuilt. In this research, an urban design case located in Tiexi District, Shenyang, China was experimented to verify our approach. To simplify the calculation, the site only consists of one block (in blue) which is an old industrial area of about 60000 m^2 (shown in Fig. 7).

4 Results

In this research, the agent was trained using co-simulation with MATLAB and Rhino/Grasshopper. MATLAB was used to code the algorithm and Rhino/Grasshopper were used to establish the model and simulate the direct sunlight hours, solar heat gains, etc. After about 150 h of training (2179 episodes, Intel(R) Core (TM) i7-7700HQ CPU @ 2.80 GHz), the agent finally generated a series of alternatives. According to the results shown in Fig. 8, there was an upward trend from Episode 1 to Episode 2000. The last group at the lower right corner was manual adjusted according to Episode 1936 which had the highest reward according to Formula (1) among all the episodes.



Episode 1

Episode 250

Episode 500

Episode 750

Episode 1000

Episode 1250

Episode 1500

Episode 1750

Episode 1936

Final scheme

Fig. 8. Alternatives generated by the approach

According to the results, the agent performed better and better during the training process. A better agent is expected to be presented in the future by being trained to better action parameters.

5 Conclusions and Future Work

The generative design approach proposed in this research is a performance-based automatic urban design approach using DRL and computer vision. Compared with conventional approaches using optimization algorithms, this method is not limited by the number of the design variables thus can generate a scheme with any numbers of buildings in any shape. The DDPG agent was trained using co-simulation with MATLAB and Rhino/Grasshopper, and Ladybug was used to simulate direct sunlight hours and solar heat gains. Although the agent may need further training, this experiment proved the feasibility of the theory. The contribution of this research lies in the advancement and demonstration of an innovative and complete DRL model applied to performance-based generative design. This approach can be implemented into other cases by changing the observation, action and reward.

However, the agent training process is very time-consuming and it also need tough conditions (like an appropriate reward function, actor and critic network structures) to converge. Besides, the different design conditions need different reward functions and function approximators. The design of the function approximators or network structures is not a new problem, but so far is still a research problem for further study.

Acknowledgement. This research is supported by the National Natural Science Foundation of China (Grant No. 51628803).

References

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

Duda, R.O., Hart, P.E.: Use of the Hough transformation to detect lines and curves in pictures. Commun. ACM **15**(1), 11–15 (1972)

Eilouti, B.: Shape grammars as a reverse engineering method for the morphogenesis of architectural facade design. Front. Architect. Res. **8**(2), 191–200 (2019)

Frazer, J.: Chapter 9 - Creative design and the generative evolutionary paradigm. In: Bentley, P.J., Corne, D.W. (eds.) Creative Evolutionary Systems, pp. 253–274. Morgan Kaufmann, San Francisco (2002)

Lambe, N.R., Dongre, A.R.: A shape grammar approach to contextual design: a case study of the Pol houses of Ahmedabad, India. Environ. Plan. B-Urban Anal. City Sci. **46**(5), 845–861 (2019)

Lee, J.H., Ostwald, M.J., Gu, N.: A Justified Plan Graph (JPG) grammar approach to identifying spatial design patterns in an architectural style. Environ. Plan. B-Urban Anal. City Sci. **45**(1), 67–89 (2018)

Li, C., Jiang, L., Sun, F.R., Zhang, K.: Generating circulation designs using shape grammars. Tsinghua Sci. Technol. **23**(6), 680–689 (2018)

Lillicrap, T.P., Hunt, J.J., Pritzel, A., Heess, N., Erez, T., Tassa, Y., Silver, D., Wierstra, D.: Continuous control with deep reinforcement learning (2015)

Ozdemir, S., Ozdemir, Y.: Prioritizing store plan alternatives produced with shape grammar using multi-criteria decision-making techniques. Environ. Plan. B-Urban Anal. City Sci. **45**(4), 751–771 (2018)

Rodrigues, E., Fernandes, M.S., Gomes, A., Gaspar, A.R., Costa, J.J.: Performance-based design of multi-story buildings for a sustainable urban environment: a case study. Renew. Sustain. Energy Rev. **113**, 109243 (2019)

Rodrigues, E., Soares, N., Fernandes, M.S., Gaspar, A.R., Gomes, A., Costa, J.J.: An integrated energy performance-driven generative design methodology to foster modular lightweight steel framed dwellings in hot climates. Energy. Sustain. Dev. **44**, 21–36 (2018)

Sun, C., Rao, J.: Study on performance-oriented generation of urban block models. Springer, Singapore (2020)

Yavuz, E., Colakoglu, B., Aktas, B.: From pattern making to acoustic panel making utilizing shape grammars. Brussels, Ecaade-Education & Research Computer Aided Architectural Design Europe (2018)

Youssef, A.M.A., Zhai, Z.Q., Reffat, R.M.: Generating proper building envelopes for photovoltaics integration with shape grammar theory. Energy Build. **158**, 326–341 (2018)