Deep Reinforcement Learning for Controlling the Groundwater in Slopes

Aynaz Biniyaz, S.M.ASCE[1]; Behnam Azmoon, S.M.ASCE[2];
and Zhen Liu, Ph.D., P.E., M.ASCE[3]

[1]Graduate Research Assistant, Dept. of Civil and Environmental Engineering, Michigan Tech Univ., Houghton, MI. Email: abiniyaz@mtu.edu
[2]Graduate Research Assistant, Dept. of Civil and Environmental Engineering, Michigan Tech Univ., Houghton, MI. Email: bazmoon@mtu.edu
[3]Associate Professor, Dept. of Civil and Environmental Engineering, Michigan Tech Univ., Houghton, MI (corresponding author). Email: zhenl@mtu.edu

ABSTRACT

Extreme weather events are a main cause of landslides in recent years. Real-time control of groundwater tables in slopes can help protect earth slopes in areas prone to flooding. Though subsurface drainage wells equipped with a pumping system is an efficient way to lower the groundwater, it has been mostly employed in short-term projects due to the high-operational costs in labor and energy. To reduce these operational costs, this paper investigates the idea of an autonomous pumping system enabled by Deep Reinforcement Learning (DRL), which is a subfield of machine learning for automated decision-making. Such a system can dynamically adapt its response to rainfall events by controlling the pumping flow rate, and more importantly, can improve the pumping policy over time. To prove the idea of the autonomous pumping system, a seepage analysis model was implemented using a partial differential equation solver, FEniCS, to simulate a lab-scale geo-system, that is, a slope equipped with a pump and subjected to rainfall events, which served as the virtual environment for the reinforcement learning. A Deep Q-learning Network (DQN), that is, a DRL agent, was implemented to learn the optimal control policy based on the trial and error process of the system to achieve the desired objective. This agent was trained to learn how to control the pump's flow rate to keep the groundwater close to the target level during different rainfall events. A reward function was defined to evaluate the state of the groundwater, which could affect the next action taken by the agent. The goal of the DQN is to find a policy that maximizes the received reward. The training was carried out from scratch without human interventions. Aiming at binary control, the agent learns whether to turn on/off the pump based on the rewards constructed with the distance of the water at the time of decision and the target level. The results showed that a DRL can learn how to control a pump to maintain the water level in a binary control mode, which may point out a promising direction for establishing intelligent geo-systems. Such autonomous control of groundwater can help mitigate landslide hazards as a long-term geotechnical solution.

INTRODUCTION

In recent years, extreme weather events, as a result of climate change, have triggered many landslides (Kirschbaum et al., 2020; Kristo et al., 2017). Prolonged intense rainfalls increase the groundwater table in slopes which can reduce the stability of the slopes. Furthermore, a higher saturation degree of the soil above the groundwater increases the slope's total weight and

decreases the shear strength of the soil, which can adversely impact the safety of such geo-systems (Alsubal et al., 2018; Cho, 2016; Sun et al., 2015).

To mitigate the hazard of landslides in areas prone to flooding, subsurface drainage systems have been widely employed to lower the groundwater level and increase the stability of slopes (Nicholson, 2014; Turner and Schuster, 1996). Drainage systems are designed to use gravity to take away the water from the slope. However, in the absence of adequate gravity forces, pumping systems are required to remove the water from deep drainage wells (Holtz and Schuster, 1996). The downside of such systems is that they require high operational costs such as labor and energy costs, which is the main reason for these systems to be utilized as short-term solutions for dewatering (Cashman and Preene, 2001). To reduce operational costs and improve the safety of slopes, this study aims to investigate the idea of using an autonomous pumping system enabled by Deep Reinforcement Learning (DRL).

Due to the introduction of DRL, a subfield of machine learning, autonomous systems have recently attracted a lot of interest. In such systems, a DRL agent learns from interactions with an environment without comprehensive knowledge about it before learning. The DRL agent takes actions aiming to maximize the cumulative reward received from the environment. In other words, the DRL agent tries different control policies to find the optimal one which returns the maximum reward (Sutton and Barto, 1998). DRL has been showing promising results in robotics (Lillicrap et al., 2015), Atari games (Mnih et al., 2015), superhuman level of play in chess (Silver et al., 2017), Google's Alpha Go (Russell and Norvig, 2002), and autonomous driving (Sallab et al., 2017). The advantage of the DRL is that it continuously learns from its experience and improves the adopted control policy.

To prove the idea of the autonomous pumping system, this study uses a seepage analysis model to simulate a lab-scale geo-system, i.e., a slope equipped with a pump and subjected to rainfall events, which served as the virtual environment for the DRL. The seepage analysis was implemented using the DOLFIN package which is a Python interface of FEniCS, a Partial Differential Equation (PDE) solver. The autonomous pumping system can dynamically modify its response to changing rainfall intensities. The major advantage of such a system is that it can improve the adopted pumping policy over time. To learn the optimal control policy and maximize the cumulative reward, a Deep Q-learning Network (DQN), i.e., a DRL agent, was implemented. The agent was trained to learn how to control the pump's flow rate during rainfall events with different patterns. The objective of the system is to keep the water table in the slope close to the target level.

The remainder of the paper is laid out as follows. First, the DRL framework and the DQN are introduced. Then, the seepage model, which simulates the DRL environment, is explained. Subsequently, network training and evaluation are provided. Finally, the results are presented, followed by conclusions.

METHODOLOGY

Reinforcement Learning for Geo-system

The objective of the intended geo-system is to control the pump's flow rate to keep the groundwater close to the target level during different rainfall events. To enable such control of the groundwater table in the geo-system (i.e., a slope equipped with a pump and subjected to rainfall events), the interactions between the DRL agent and environment, i.e., a slope equipped

with a pump and subjected to rainfall events, were formulated in a formal framework called Markov Decision Process (MDP) (Mason and Grijalva, 2019). Figure 1 shows the interactions between the DRL agent and the geo-system environment. In this environment, at any given time *t*, the DRL agent receives the state S_t (i.e., the water head at point "P") in the slope from the conducted seepage analysis and then takes a control action A_t (i.e., turning on or off the pump). Consequently, the agent receives a reward R_t (i.e., evaluation score for the taken action based on the state of groundwater) from the environment for the choice of action. The reward is formulated to score the performance of the DRL agent in controlling the water table.

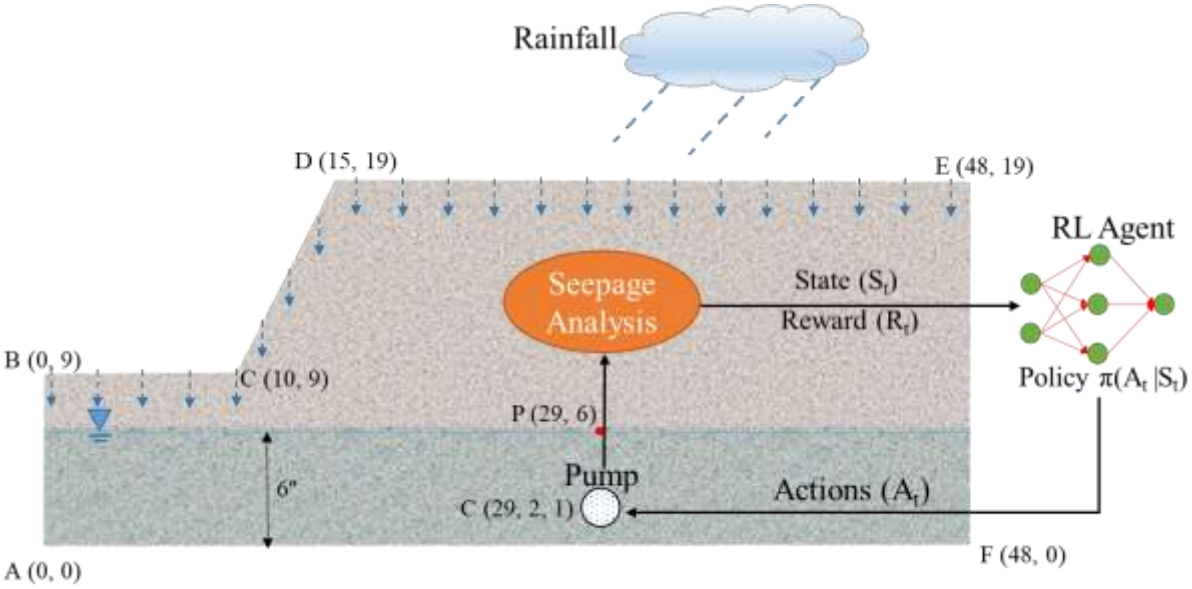

Figure 1. Learning loop of a DRL agent in the lab-scale geo-system (units: inch)

The DRL agent will receive a positive reward if the groundwater gets closer to the target level and will be penalized if the groundwater moves away (up or down) from the target level. The target level is assumed to be the water level at point "P" (which could be selected differently for other geo-systems to maintain the safety of the slope) as shown in Figure 1. The agent makes decisions regarding the actions to take and learns which action will be better in a specific state based on the reward received from the environment.

A Deep Q-learning Network (DQN), first introduced by Mnih et al. (2015), was implemented to learn the optimal policy that can lead to the highest cumulative rewards. Integrating a Deep Neural Network (DNN) algorithm with reinforcement learning has shown to be very effective in solving problems with high-dimensional states and actions. Basically, a Q-learning algorithm assigns a state-action value (i.e., $Q(S_t, A_t)$) to a given state-action (S_t, A_t) as follows,

$$Q(S_t, A_t) \leftarrow (1-\alpha) Q(S_t, A_t) + \alpha \left[R_t + \gamma \max Q(S_{t+1}, A_{t+1}) \right], \quad (1)$$

where R_t is the reward in response to the taken action A_t in the state S_t, $\max Q(S_{t+1}, A_{t+1})$ is the maximum Q-value in the next state S_{t+1} after taking the (optimum) action A_{t+1}, α is the learning

rate of the agent, and γ is the discount factor which manages the contribution of the long-term rewards from future states and actions. Since Eq. 1 is an iterative equation, the Q-values will be initialized with arbitrary values (usually with zeros). By taking the action, the Q-value will be continuously updated until converging to an optimal policy. As the number of states and actions increases, a DQN algorithm can efficiently approximate the Q-values.

The epsilon-greedy strategy (Li et al., 2010) was adopted for the selection of actions. Such a strategy helps reach a balance between gaining more rewards by using exploitation (by far the most effective action discovered by the agent) and exploration (exploring a possibly better action) (Cohen et al., 2007). In fact, the DRL agent takes an action associated with the highest Q-value with the possibility of $(1-\varepsilon)$ and chooses a random action with the possibility of ε (see Eq. 7). Initially, ε is set to a high value (e.g., 1) to allow the agent to explore more for efficient actions, and is gradually reduced to a lower value (e.g., 0.01) to allow the agent to exploit more as the agent achieves a rigorous control policy (Sun, 2020).

$$\text{Action at time t, } A_t \begin{cases} \text{action associated with } \max Q(S_t, A_t), & \text{with possibily 1-}\varepsilon \\ \text{random action,} & \text{with possibily } \varepsilon \end{cases}, \quad (2)$$

The loss function is the mean square error of the predicted Q-value and the target Q-value, Eq. 3. Here, $\left(R_t + \gamma \max Q\left(S_{t+1}, A_{t+1}\right)\right)$ is the target Q-value and $Q\left(S_t, A_t\right)$ is the predicted Q-value.

$$Loss = \left[\left(R_t + \gamma \max Q\left(S_{t+1}, A_{t+1}\right)\right) - Q\left(S_t, A_t\right)\right]^2 \quad (3)$$

Seepage Model

In this section, the seepage model that served as the DRL environment is explained. A transient seepage analysis was conducted to update the DRL agent on the state S_t (i.e., the water head at point "P" in Figure 1) at any given time t. The seepage analysis was implemented using the DOLFIN package (Logg and Wells, 2010), which is a Python interface of FEniCS (Alnæs et al., 2015). FEniCS is an open-source platform for solving Partial Differential Equations (PDEs). The computational framework for this model was validated by commercial software (Biniyaz et al., 2021). The governing equation for the transient seepage model was obtained by modifying the Richards equation (Liu, 2018).

$$S\frac{\partial(h+z)}{\partial t} = K \times \nabla\left(\nabla\left(h+z\right)\right) + q_s, \quad (4)$$

where h [m] is the pressure head, z [m] is the elevation head, q_s [m/s] is the sink/source term representing the applied flux for the pump. For a saturated flow, S and K were replaced with K_s (i.e., saturated hydraulic conductivity) and S_s (i.e., specific storage of saturated flow), respectively. For the unsaturated flow, these parameters were replaced with $K_s K_r$ (K_r is the relative hydraulic conductivity) and S_c (i.e., specific moisture content), respectively. Relative

hydraulic conductivity defines how the hydraulic conductivity changes with the effective saturation degree (S_e). The following equation proposed by van Genuchten (1980) was adopted for formulating K_r:

$$K_r = S_e^b \left(1-\left(1-S_e^{1/a}\right)^a\right)^2,\ S_e = \left[1+\left(\frac{\psi}{P_0}\right)^{\frac{1}{1-a}}\right]^{-a}, \tag{5}$$

where a, b, and P_0 [Pa] are fitting parameters derived from Soil-Water Characteristic Curve (SWCC), S_e is the effective saturation degree, and ψ [Pa] is the matric suction (Cho, 2016). S_c is the derivative of volumetric water content (θ) with respect to the h:

$$S_c = \left|\frac{\partial\theta}{\partial h}\right| = n\frac{\partial S_e}{\partial h} = \frac{na}{h(a-1)}\left(1+\left(\frac{\psi}{P_0}\right)^{\frac{1}{1-a}}\right)^{-a-1}\left(\frac{\psi}{P_0}\right)^{\frac{1}{1-a}}, \tag{6}$$

where n [-] is the soil porosity.

A lab-scale slope profile equipped with a pump was used in the analyses as shown in Figure 1. Since the lab-scale geosystem was located in a tank, it was assumed that there is no in-flux/out-flux from left, right, and bottom of the slop, and rain infiltration is the only in-flux through the slop's surface. The pump was modeled as a sinkhole with a radius of 1 inch. The initial groundwater table was set to the level of 6 inches.

The no-flux boundary condition was applied to the left, right, and bottom of the slope, i.e., "AB" and "EF", and "FA". The slope's surface boundaries, i.e., "BC", "CD", and "DE", were set to the in-flux boundary conditions representing the rainfall infiltration. It is noted that the water ponding was not considered for the slope's surface since the rain intensities (I_r) are mostly less than the soil infiltration capacity. Change in pore air pressure was also ignored in this study.

Four rainfall events with different durations, patterns, and depths of precipitation were considered in this study as shown in Figure 2. Figure 2a shows a 15 minutes-rainfall event with constant intensity over time. Figure 2b shows a 15 minutes-rainfall event with a normal distribution (i.e., maximum rain intensity happens in the middle of the event). Figure 2c shows a 20 minutes-rainfall event with descending pattern (i.e., maximum rain intensity happens at the beginning of the event). Figure 2d shows a 25 minutes-event with ascending pattern (i.e., the maximum rain intensity happens at the end of the event). The out-flux boundary was assigned to the pump as follows,

$$-\nabla(h+z)\cdot\vec{n} = \frac{Q_p}{2\pi r} \quad \text{on } \Gamma_{(\text{Pump})} \text{ for } t>0, \tag{7}$$

where Q_p is the full capacity of the pump which was considered 0.0002 m^3/s (3.17 Gallon Per Minute, GPM) and r is the radius of the sinkhole for the pump. Soil properties for the seepage model were also presented in Table 1. It is noted that these values were assumed based on the ranges reported in Cho (2016) and Sethi and Di Molfetta (2019) for sand.

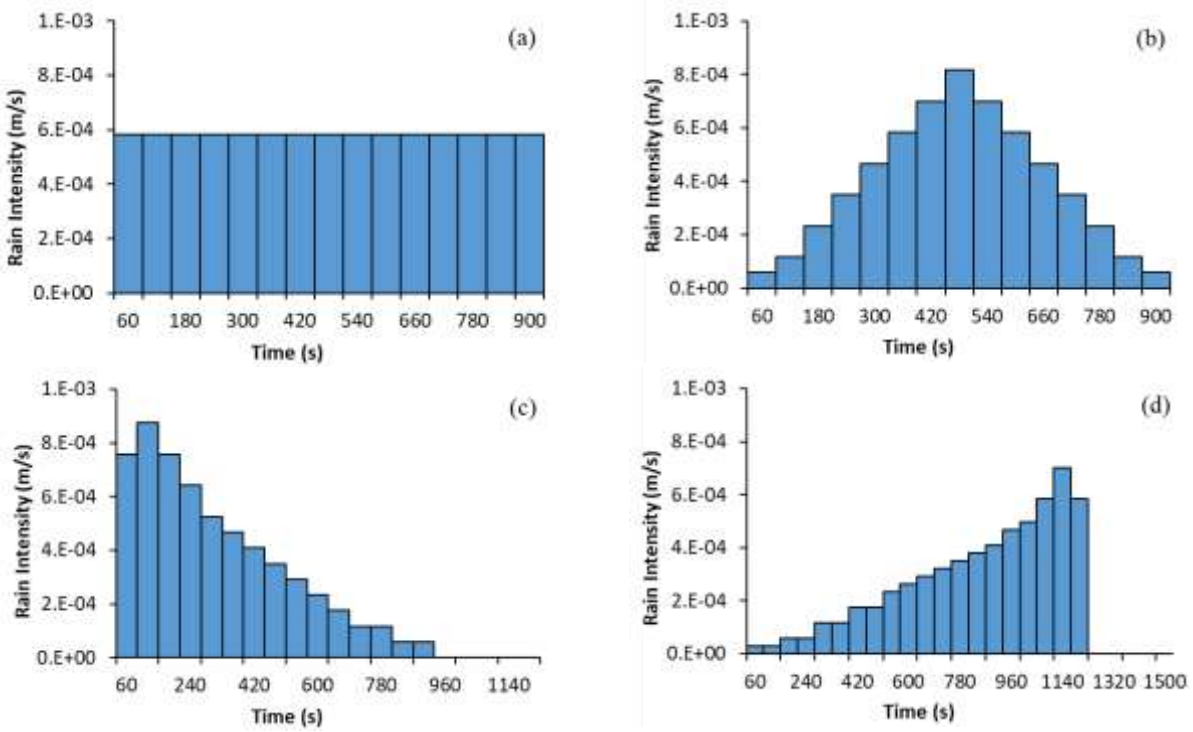

Figure 2. Rainfall events with four different patterns and duration: (a) 15 minutes-event with constant intensity over time, (b) 15 minutes-event with maximum intensity at the middle of precipitation (c) 20 minutes-event with maximum intensity at the beginning of precipitation, and (d) 25 minutes-event with maximum intensity at the end of precipitation

Table 1. Soil properties for the seepage model adopted from Cho (2016) and Sethi and Di Molfetta (2019)

Model input	Definition	Sand
K_s	Saturated hydraulic conductivity (m/s)	6×10^{-4}
S_s	Saturated specific storage (1/m)	1×10^{-4}
n	Porosity	0.32
P_0	Empirical parameter (Pa)	1200
a	Empirical parameter	0.6
b	Empirical parameter	0.5

Network Training and Evaluation

A Deep Neural Network was implemented to learn the action-value function and approximate the Q-values. The network consists of two hidden layers with 25 neurons per layer and one output layer with 2 neurons. Fully connected dense layers were used for all three layers in the network's architecture. The output of the network was the Q-values associate with two possible actions for the agent (i.e., turning on or off the pump). A ReLu activation (Goodfellow et al., 2016) function was used for the hidden layers and a sigmoid activation function (Elfwing

et al., 2018) was used for the output layer. The network parameters used in the study were presented in Table 2. These parameters were achieved by optimizing the network architecture. It is noted that one rainfall event was used to train the agent at a time.

Table 2. Optimized model parameters for DQN

Parameter	Value
Number of episodes for training	300
Batch size	2
Learning rate, α	1e-3
Gamma, γ	0.95
Initial epsilon	1
Final epsilon	0.01
Epsilon decay	0.995

To train the DRL agent and evaluate the action A_t taken by the agent in the state S_t, a reward function was formulated as Eq. 8. Such an explicit reward function gives better guidance to the DRL agent to achieve the goal of the system (Mullapudi et al., 2020). This reward function was constructed based on the water head at point "P" in Figure 1 at the time $t+1$ and time t.

$$R_t\left(A_t, S_t\right) = if \begin{cases} \left|h_p(t+1)\right| \le \left|h_p(t)\right|, & \left|h_p(t+1) - h_p(t)\right| * 1000 \\ h_p(t+1) > h_p(t) \text{ \& action=Pump on}, & \left(h_p(t+1) - h_p(t)\right) * 1000 \\ h_p(t+1) \le -5", & \left(h_p(t+1)\right) * 100 \\ h_p(t+1) \ge 13", & \left(-h_p(t+1)\right) * 100 \\ \text{else}, & -\left|h_p(t+1) - h_p(t)\right| * 100 \end{cases} \tag{8}$$

where $h_p(t)$ is the water head at point "P" and at the time t, $h_p(t+1)$ is the water head at point "P" and at the time t+1. The water head can take positive and negative values. A positive and negative value of h_p indicates that the water level is above and below the point "P", respectively. The absolute value of the total head at point "P" represents the distance from point "P", (i.e., target level). Comparing the water head at the time t (i.e., before taking the action A_t) and time t+1 (i.e., after taking the action A_t) helped evaluate the agent's action. Based on this reward function, the agent received a positive reward for reducing the distance from the target level. By contrast, the DRL agent was penalized in case of overflow (i.e., the height of water above point "P" is more than 13 inches), discharge (i.e., the drop in the water table is more than 5 inches), or increasing the distance from the target level.

RESULTS

After the training process was completed, the performance of the DRL agent in controlling the groundwater was evaluated. Figure 3a, Figure 3b, Figure 3c, and Figure 3d display the

distance from the target level (water level at point "P") during the four different rainfall events shown in Figure 2a, Figure 2b, Figure 2c, and Figure 2d, respectively. In fact, the water head at point "P" in Figure 1, which represents the distance of the water table from this point, was plotted during the rainfall events to evaluate the agent's performance. As shown in Figure 3, in all four rainfall events with varying patterns, durations, and precipitation depths, the DRL agent used a distinct combination of actions (pumping on and off) to keep the water table close to the target level (i.e., 6 inches). A comparison of the graphs indicates that the agent effectively learned the optimal control policy to manage the groundwater under different weather conditions.

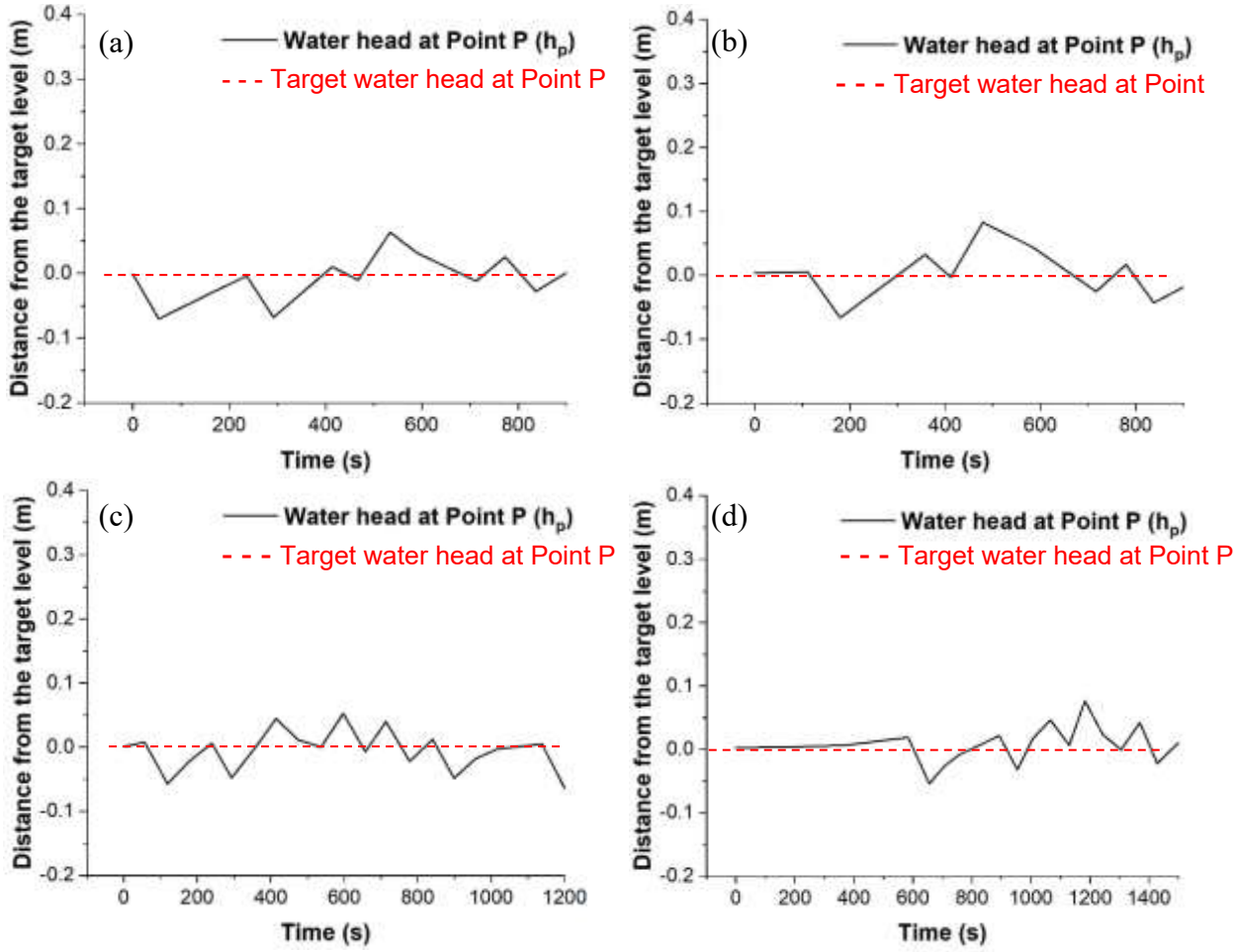

Figure 3. Control of the groundwater by the DRL agent during different rainfall events: (a) 15 minutes' event with constant intensity over time, (b) 15 minutes' event with maximum intensity at the middle of precipitation (c) 20 minutes' event with maximum intensity at the beginning of precipitation, and (d) 25 minutes' event with maximum intensity at the end of precipitation

Additionally, Figure 3 confirmed that the DRL agent improves the control policy over time. The DRL agent was first trained with 15 minutes-rainfall event with constant rain intensity shown in Figure 2a. Then, the agent was trained with 15 minutes-rainfall event with normal

distribution, 20 minutes-rainfall event, and 25 minutes rainfall events shown in Figure 2b, Figure 2c, and Figure 2d, respectively. By moving from Figure 3a toward Figure 3d, it can be observed that the distance from the target level was reduced as the agent gained more experience in controlling the water table under different rainfall events.

CONCLUSION

This study aimed to take a small but significant step toward developing an autonomous pumping system and minimizing the operational costs of groundwater control. The main contribution of this paper is the implementation of Deep Reinforcement Learning for real-time control of groundwater in slopes. The results showed that the implemented DQN agent can learn how to control a pump to lower the water table and mitigate the landslide in the slope. Despite the varying rainfall pattern, duration, and precipitation depth, the DRL agent could successfully learn the effective control policy to keep the water level near the target level. The findings of this study point to a feasible avenue for developing intelligent geo-systems.

REFERENCES

Alnæs, M., Blechta, J., Hake, J., Johansson, A., Kehlet, B., Logg, A., Richardson, C., Ring, J., Rognes, M. E., and Wells, G. N. (2015). "The FEniCS project version 1.5." *Archive of Numerical Software*, 3(100).

Alsubal, S., Sapari, N., and Harahap, S. (2018). "The Rise of groundwater due to rainfall and the control of landslide by zero-energy groundwater withdrawal system." *International journal of engineering & technology*, 7(6).

Biniyaz, A., Azmoon, B., and Liu, Z. (2021). "Coupled transient saturated–unsaturated seepage and limit equilibrium analysis for slopes: influence of rapid water level changes." *Acta Geotechnica*, 1-18.

Cashman, P. M., and Preene, M. (2001). *Groundwater lowering in construction: A practical guide*, CRC Press.

Cho, S. E. (2016). "Stability analysis of unsaturated soil slopes considering water-air flow caused by rainfall infiltration." *Engineering geology*, 211, 184-197.

Cohen, J. D., McClure, S. M., and Angela, J. Y. (2007). "Should I stay or should I go? How the human brain manages the trade-off between exploitation and exploration." *Philosophical Transactions of the Royal Society of London B: Biological Sciences*, 362(1481), 933-942.

Elfwing, S., Uchibe, E., and Doya, K. (2018). "Sigmoid-weighted linear units for neural network function approximation in reinforcement learning." *Neural Networks*, 107, 3-11.

Goodfellow, I., Bengio, Y., and Courville, A. (2016). *Deep learning*, MIT press.

Holtz, R. D., and Schuster, R. L. (1996). "Landslides: Investigation and Mitigation." *Transportation Research Board, Special Report*, 247, 439-473.

Kirschbaum, D., Kapnick, S., Stanley, T., and Pascale, S. (2020). "Changes in extreme precipitation and landslides over High Mountain Asia." *Geophysical research letters*.

Kristo, C., Rahardjo, H., and Satyanaga, A. (2017). "Effect of variations in rainfall intensity on slope stability in Singapore." *International soil and water conservation research*, 5(4), 258-264.

Li, W., Wang, X., Zhang, R., Cui, Y., Mao, J., and Jin, R. "Exploitation and exploration in a performance based contextual advertising system." *Proc., Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, 27-36.
Lillicrap, T. P., Hunt, J. J., Pritzel, A., Heess, N., Erez, T., Tassa, Y., Silver, D., and Wierstra, D. (2015). "Continuous control with deep reinforcement learning." arXiv preprint arXiv:1509.02971.
Liu, Z. L. (2018). *Multiphysics in Porous Materials*. Multiphysics in porous materials, Springer, 29-34.
Logg, A., and Wells, G. N. (2010). "DOLFIN: Automated finite element computing." *ACM Transactions on Mathematical Software (TOMS)*, 37(2), 1-28.
Mason, K., and Grijalva, S. (2019). "A review of reinforcement learning for autonomous building energy management." *Computers & Electrical Engineering*, 78, 300-312.
Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M., Fidjeland, A. K., and Ostrovski, G. (2015). "Human-level control through deep reinforcement learning." *nature*, 518(7540), 529-533.
Mullapudi, A., Lewis, M. J., Gruden, C. L., and Kerkez, B. (2020). "Deep reinforcement learning for the real time control of stormwater systems." *Advances in Water Resources*, 140, 103600.
Nicholson, P. G. (2014). *Soil improvement and ground modification methods*, Butterworth-Heinemann.
Russell, S., and Norvig, P. (2002). *Artificial intelligence: a modern approach*.
Sallab, A. E., Abdou, M., Perot, E., and Yogamani, S. (2017). "Deep reinforcement learning framework for autonomous driving." *Electronic Imaging*, 2017(19), 70-76.
Sethi, R., and Di Molfetta, A. (2019). *Groundwater Engineering: A Technical Approach to Hydrogeology, Contaminant Transport and Groundwater Remediation*, Springer International Publishing.
Silver, D., Hubert, T., Schrittwieser, J., Antonoglou, I., Lai, M., Guez, A., Lanctot, M., Sifre, L., Kumaran, D., and Graepel, T. (2017). "Mastering chess and shogi by self-play with a general reinforcement learning algorithm." arXiv preprint arXiv:1712.01815.
Sun, A. Y. (2020). "Optimal carbon storage reservoir management through deep reinforcement learning." *Applied Energy*, 278, 115660.
Sun, D.-M., Zang, Y.-G., and Semprich, S. (2015). "Effects of airflow induced by rainfall infiltration on unsaturated soil slope stability." *Transport in porous media*, 107(3), 821-841.
Sutton, R. S., and Barto, A. G. (1998). *Reinforcement learning: An introduction*, MIT press.
Turner, A. K., and Schuster, R. L. (1996). "Landslides: investigation and mitigation. Special Report 247." Trans. Res. Board, national academy press, Washington, DC.
van Genuchten, M. T. (1980). "A closed-form equation for predicting the hydraulic conductivity of unsaturated soils 1." *Soil science society of America journal*, 44(5), 892-898.