# Reinforcement learning-driven decision support for target-oriented branch pruning on urban trees

Qiguan Shu, Kai Zhe Boey and Ferdinand Ludwig
*School of Engineering and Design, Technical University of Munich, Munich, Germany*

Received 11 October 2024
Revised 20 February 2025
13 March 2025
Accepted 20 March 2025

## Abstract

**Purpose** – The conventional design and management of urban trees often overlook the benefits of specific canopy shapes, despite their crucial role in enhancing thermal comfort and optimizing direct sunlight utilization. This study presents a novel workflow in which designers define target leaf areas, and a decision-support algorithm guides tree management specialists in regulating growth through branch pruning to meet these targets.
**Design/methodology/approach** – We developed a framework that integrates a tree growth simulation game with a deep reinforcement learning (DRL) network for decision-making. The simulation predicts growth responses to pruning and assesses how closely the resulting structure matches the target leaf area. Based on the current tree state and reward feedback, the DRL network issues pruning decisions. The DRL network learns to optimize pruning strategies by iteratively interacting with the simulation game.
**Findings** – The configured network proved effective in navigating the complex and extensive hybrid decision space associated with tree pruning. It successfully acquired techniques to minimize penalties and consistently achieve relatively high reward scores in the game.
**Research limitations/implications** – High computational resource consumption remains a significant challenge. Additionally, the reward function lacks clear definitions that consistently guide the model toward the intended design targets.
**Originality/value** – This work establishes a novel technical pathway for implementing the proposed workflow, employing a voxel approach in the design and management of urban trees. It facilitates multifunctional tree use aligned with explicitly defined design objectives.

**Keywords** Computational design, Tree information modeling, Reinforcement learning, Quantitative structure model for trees, Voxel approach in design, Branch pruning

**Paper type** Research paper

## Nomenclature

| | |
|---|---|
| $S_t$ | The state of the agent at round t |
| $A_t$ | The action taken at round t |



*Erratum:* It has come to the attention of the publisher that the article, Shu, Q., Boey, K.Z. and Ludwig, F. (2025), "Reinforcement learning-driven decision support for target-oriented branch pruning on urban trees", *Smart and Sustainable Built Environment*, Vol. ahead-of-print No. ahead-of-print. https://doi.org/10.1108/SASBE-10-2024-0427 originally published Figures A1, A2 and A3 in the 'Figures' section and Figures 1, 2 and 3 in the 'Appendix' section in the online version. This has now been rectified. The publisher sincerely apologises for this error and for any inconvenience caused.

Thanks to Arne Hingst and Gehard Schubert for Coordinating the use of the LRZ Cluster computer. This resource is crucial in training the proposed reinforcement learning networks. We would also like to show our gratitude to our research partners in the DFG-DACH project: Halil Erdal, Thomas Rötzer, Astrid Reischl, Hans Pretzsch, Michael Hensel, Jakub Marcin, and Aljbin Ahmeti.

*Conflicts of interest:* The authors declare no conflict of interest.

*Funding:* This study was funded by the DFG-DACH project named Urban Green System 4.0 (grant number LU2505/2-1 AOBJ:683826 42).

*Data availability:* The data underlying this article are available in [Branch-Pruning-Game-on-Urban-Trees], at https://github.com/QiguanShu/Branch-Pruning-Game-on-Urban-Trees.

Smart and Sustainable Built Environment
Vol. 15 No. 5, 2026
pp. 1917-1938
Emerald Publishing Limited
e-ISSN: 2046-6102
p-ISSN: 2046-6099
DOI 10.1108/SASBE-10-2024-0427

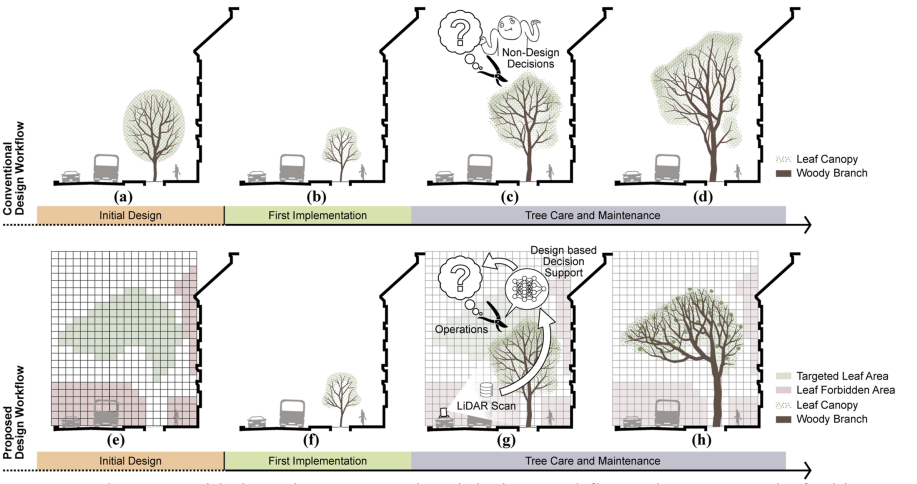
| | |
|---|---|
| $R_t$ | The reward at round t |
| $Q(S_t, A_t)$ | The Q-value given a state and an action |
| $k \in K$ | Discrete action types |
| $x_k \in \mathcal{M}_k$ | Continuous parameter corresponding to an action type |
| $\pi$ | Policy for decision making |
| $\varepsilon$ | Probability of using a random action |

## 1. Introduction

In light of global warming and the urban heat island effect (Rahman *et al*., 2020; Wong *et al*., 2021), there is an increasing need for diverse and multifunctional use of urban trees. Studies have shown that the effect of trees in providing thermal comfort is directly related to their particular crown structure and geometry (Krayenhoff *et al*., 2014; Oshio *et al*., 2021). Recent practices in landscape architecture have also experimented with tree planting designs to optimize the period and intensity of indoor sunlight through seasons (Pan and Jakubiec, 2022). A larger crown can better shade the pedestrians and the building façade to reduce extreme sun radiation exposure in summer (Palme *et al*., 2019). Trees' crown shapes, in this case, must be site-specific and carefully designed to offer such add-on ecosystem services.

In a traditional urban tree design and management workflow, trees are drawn by landscape architects. Their canopies are commonly illustrated in perfect geometrical forms with a fixed size (see Figure 1a). A young tree newly transplanted from a tree nursery to the site will not reach this size. With limited biomass, it cannot provide many add-on functions (see Figure 1b). For a longer period of tree growth later, arborists (tree management specialists), instead of designers, work hands-on to take care of the trees' safety and health (see Figure 1c). Therefore, the outcome of urban trees follows mostly rules such as keeping certain heights above vehicle lanes and a certain distance to buildings (see Figure 1d).

However, to guide them in providing more ecosystem services, tree crowns must be strategically managed, especially pruned, to approach a design goal. The performance of trees




**Note(s):** The upper side is under a conventional design workflow. The tree was drafted in a fixed size, and the tree management follows safety rules. The lower side is a Target-Oriented tree management workflow with the help of a decision support algorithm. Designers design a target leaf area with voxels, and tree specialists follow advice from a neural network to guide the tree growth to approach this target

**Source(s):** Authors' own work

**Figure 1.** Scenarios of a possible street tree case

can be quantified from multiple perspectives, such as improving air quality (Vos *et al*., 2013) and maximizing cooling effects (Grylls and van Reeuwijk, 2021). These simulation tools can also help optimize targets of leaf areas. Designers or decision-makers can work with voxels to plan the targeted crown shapes and density in the 3-D space (Ludwig *et al*., 2024; Yazdi *et al*., 2023) (see Figure 1e). In this novel workflow, driving the tree growth towards such design targets is key. An arborist could be advised by a decision support algorithm (see Figure 1g) at any stage of its growth. Following this advice, the outcome of urban trees can better approach what had been designed (see Figure 1h). This vision is impossible with the conventional workflow, where the designs of urban trees and tree management are carried out independently.

Bringing this vision to reality requires robust technical foundations, including (1) advanced tree sensing and modeling (Nitoslawski *et al*., 2019), (2) target-oriented decision-support algorithms for branch pruning at the single-tree level (Yazdi *et al*., 2023), and (3) the integration of tree-level decision support to align with urban-scale green infrastructure management strategies, also enabling top-down planning.

Regarding tree sensing and modeling, LiDAR has been proven efficient in sensing the trees and their environment (Abegg *et al*., 2017; Dan *et al*., 2012; Shu *et al*., 2022; Yazdi *et al*., 2024). Based on detailed scans of trees in the form of point clouds, quantitative structure models of trees (QSMs) enable detailed documentation of branch segments and their topological connections. In the context of numerous open-source methodologies, treeQSM (Raumonen *et al*., 2013) has demonstrated superior performance in terms of noise resistance and fidelity to reality when compared with AdTree (Du *et al*., 2019) and AdQSM (Fan *et al*., 2020). The collected QSM data of trees enable the prediction of resprouting patterns of new shoots following branch pruning (Shu *et al*., 2024b). Additionally, leaf area density (LAD) within voxel grids can be estimated using this data (Shu *et al*., 2024a). Despite limitations in these estimation models regarding species and dataset size at their current stages, intensive data about trees will be gained exponentially using remote sensing approaches in the foreseeable future. This also means that deeper and more knowledge of tree growth will be available to designers and decision-makers through simulation software and plug-ins sooner or later.

Regarding the decision-support algorithms for branch pruning at the single-tree level, the primary challenge lies in the vast decision space, particularly regarding time sequences. Given that the number of possible pruning scenarios grows exponentially, it is infeasible to predefine an optimal solution for every situation. Common methods in handling such complex decision space include Monte Carlo Tree Search (MCTS) (Fu, 2016), which reduces computational complexity by employing randomized sampling, and Markov Decision Processes (MDPs) (Puterman, 2014), which provides a mathematical framework in situations where outcomes are partly stochastic and partly controllable. However, both MCTS and MDPs require well-defined state boundaries—for example, a robot's position within a fixed space or the presence and color of pieces on a Go board. In contrast, tree growth lacks fixed structural rules, making defining such clear state boundaries impractical. The number, connectivity, and arrangement of branches vary dynamically, posing significant challenges for conventional decision-making models. A more suitable approach is training a neural network on a relatively large yet finite set of scenarios with explicitly defined rewards for its decisions. Through this process, this neural network is expected to self-learn and attain robust decisions in other untrained situations to achieve a high reward or minimize the penalty. Typically, such training processes use a "game" (a simulated environment) to generate different situations and rewards to guide the decision-making capability of the neural network (Diallo *et al*., 2017; Lee *et al*., 2018; Vinyals *et al*., 2017). To structure such a neural network model, classical reinforcement learning (RL) models predominantly focus on lower dimensional problems such as tabular data but often face scalability limitations when addressing complex problems such as the decisions in urban tree management. These limitations arise from their reliance on accurate value functions on every tree state (Wang *et al*., 2024). In contrast, deep neural networks perform better by approximating value functions and policies instead of exhaustively computing them for every

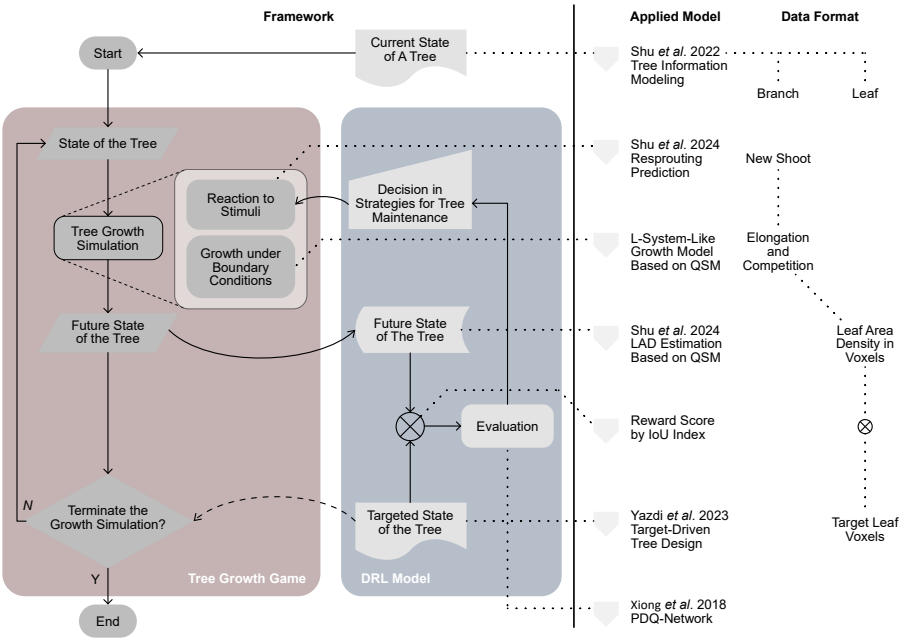
state. By integrating deep learning with RL, deep reinforcement learning (DRL) can overcome these shortcomings faced by conventional RL (Mnih *et al*., 2013). Therefore, DRL is a more suitable approach for decision-making problems in complex environments with high dimensional inputs. Due to these features, DRL has been widely applied across a variety of domains since then, including gaming (Silver *et al*., 2016; Vinyals *et al*., 2017), manufacturing (Li *et al*., 2023), and building energy management (Yu *et al*., 2021). These applications highlight the versatility and effectiveness of DRL in complex decision-making.

Regarding an urban-scale decision system for managing green infrastructure, digital twins of individual trees, including their detailed representations (Shu *et al*., 2022) and ecological interactions, serve as a fundamental foundation. Unlike traditional static approaches, a smart green infrastructure management system enables adaptive and responsive decisions (Bittencourt *et al*., 2024). For instance, urban trees are deeply interconnected with water management, influencing irrigation needs (Silva *et al*., 2020), reducing runoff, and mitigating flood risks (Dowtin *et al*., 2023). As precipitation patterns fluctuate, irrigation and rainwater harvesting decisions can be optimized by leveraging real-time data and predictive analytics to optimize resource allocation and respond to these changing conditions (Rambhia *et al*., 2023). Large-scale tree monitoring originated in forestry research (Holmgren and Thuresson, 1998), primarily for yield estimation and resource management (Helmes and Stockbridge, 2011). However, these methodologies are now being adapted for urban forestry, where monitoring focuses on tree count, height, volume, and ecological contributions (Brandt *et al*., 2025). Such information is used to inform tree planting to minimize the urban heat island effect (Francis *et al*., 2023). Despite this application, urban-scale tree monitoring has not yet transitioned into decision-making systems for tree management. Instead, pioneer research has explored using tilt-angle sensors to detect early signs of tree instability before extreme weather events, such as typhoons (Chau *et al*., 2023). Conversely, in preparation for such events, targeted pruning strategies must be implemented at the individual tree level to reduce the risk of failure and enhance public safety. This necessity has become another driver for automated decision-support algorithms for branch pruning, paving the way toward a responsive and proactive urban forestry management system.

In this context, it is crucial to establish a structured workflow for developing digital tools that can bridge the gap between tree design and management. Pruning is among the most effective methods for shaping tree crowns (Bedker *et al*., 2012). So far, pruning and growth simulations, such as EduAPPLE (Kohek *et al*., 2015), have been primarily physiology-based using the L-system (Prusinkiewicz and Lindenmayer, 1996). However, these approaches do not align with the target-oriented design envisioned in this study (see Figure 1). Therefore, this research introduces a novel objective: optimizing branch pruning decisions, marked by cylinders within the QSM of trees, to approach specific crown geometry. Following this aim, this study addresses two key questions: (1) How can such an algorithm be developed? (2) What can be achieved, and what obstacles remain in current technologies? Building on the successes of deep reinforcement learning (DRL) reported in the literature, this study investigated whether DRL could generate optimal branch manipulating decisions for individual trees to approach a design target represented by voxels.

## 2. Method

The geometric primitives of a tree can be separated for branches, leaves, and roots following Tree Information Modeling (Shu *et al*., 2022). Among the three organs, growth simulations, especially those following pruning operations, must rely on geometrical and topological representatives of branches. The voxels can describe targeted spatial areas for leaves (Yazdi *et al*., 2023). The roots are outside the scope of this study due to a lack of validated data and models nowadays. Consequently, QSM and LAD per voxels were defined as primitives for tree state in the workflow experimented below (see “Data Format” in Figure 2).




**Note(s):** Including its core theoretical framework and implementation examples. Every model in the implementation section can be replaced with other models with equivalent functions for simplification or deepened analysis

**Source(s):** Authors' own work

**Figure 2.** Overview of the workflow proposed in this study

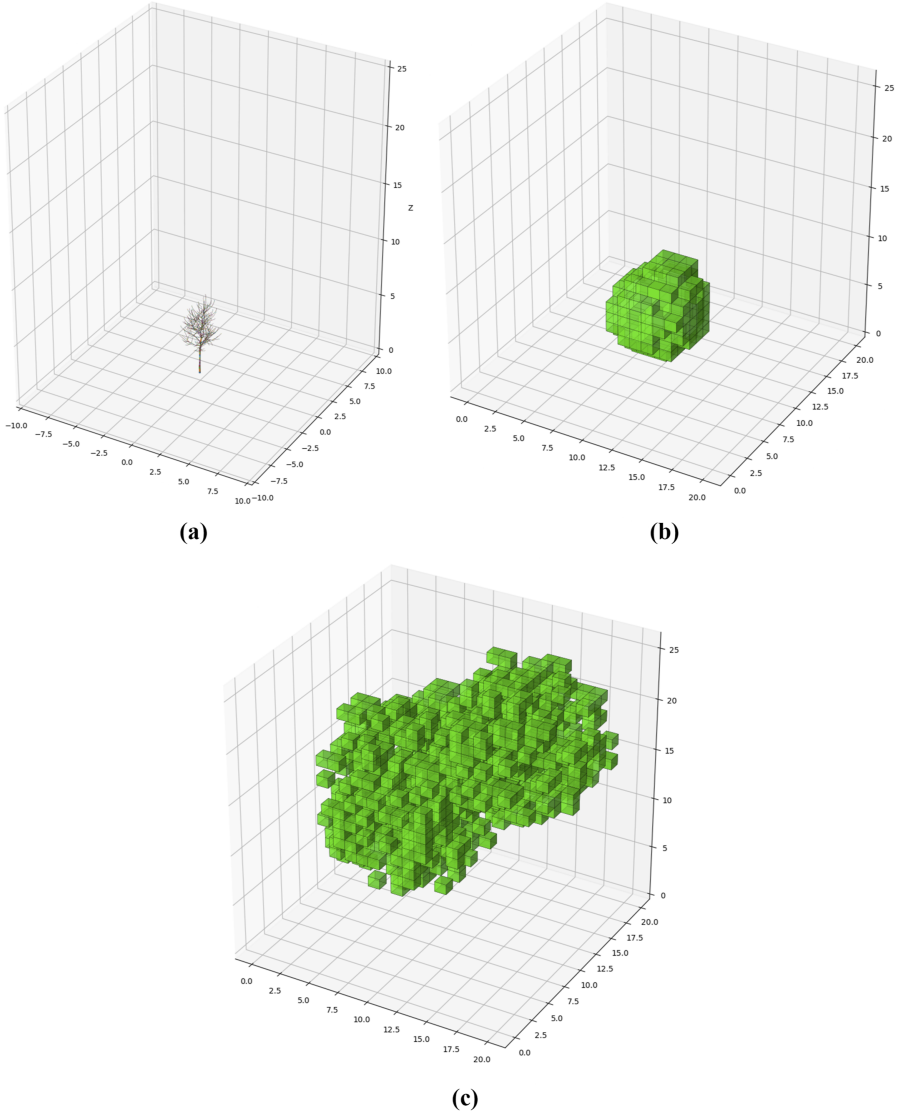
A theoretical framework was developed using the current tree state of branches in QSM and the targeted state of leaves in voxels as two initial inputs (illustrated in Figure 2 left). It was used to train a decision mechanism for pruning decisions through DRL. The framework comprises two main sectors: a tree growth game (see section 2.1) and a DRL Model (see section 2.2). The tree growth game deals with an iterative growth simulator of trees, allowing input for pruning decisions based on QSM and predicting the LAD in a future state for each turn. The DRL model is a machine player of the game. It chooses from preset pruning strategies and evaluates their effects by comparing the received LAD results from the game with the target leaf voxels.

In this workflow, each function is highly modular. Therefore, all the models in the current implementation (see "Applied Model" in Figure 2) can be replaced or revised with other models with equivalent effects.

### *2.1 The tree growth game*

As shown in the red section in Figure 2, every episode of the tree growth game starts with the initial state of a tree. In the user interface of this game, a target leaf voxel is shown next to the current leaf voxel state for pruning decisions. In our experiments, no specific design was made to set those targeted leaf voxels. Such a design is supposed to be site-specific, considering multifunctional performances, including ecosystem services and aesthetic value. How these target leaf voxels are decided is irrelevant to training the decision-making algorithm in this

study. To bypass this, a cluster of voxels was randomly populated at the upper parts of the voxel space as virtual targets for this game (see Figure 3c). For each episode, the targets are different. After setting up an initial tree state and a target state, the tree's growth consists of two simulation components: the tree's particular reaction to exogenous stimuli and a generic growth under given environmental conditions. The game's objective is to select optimal




**Source(s):** Authors' own work

**Figure 3.** Plotted tree states displayed to players in an example episode: (a) Initial branch state in QSM of a young plane tree; (b) Estimated LAD of this young plane tree at its initial state; (c) A randomly populated target LAD in the game

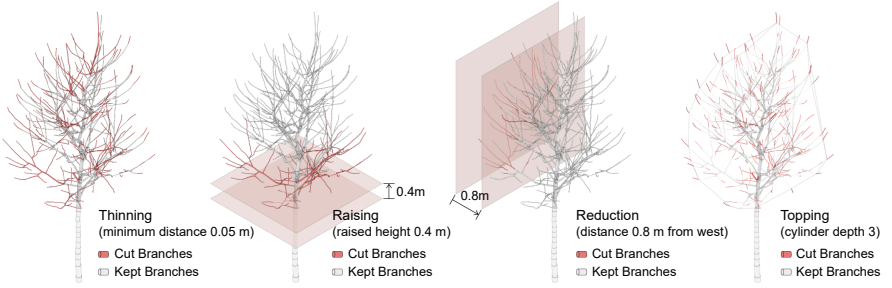
pruning strategies over a time sequence to progressively align the leaf voxel state shown in Figure 3b with the target state depicted in Figure 3c.

*2.1.1 The first tree growth game for training.* In the first training version, we randomly picked one of seven young plane trees scanned in a tree nursery (see Figure 3a) at the start of each episode. They had almost the same diameters at breast height (*DBH*), crown diameter, and height but not the same branches. Accordingly, the LAD distribution of this tree was estimated by allocating cylinders from QSM to individual voxels (Shu *et al.*, 2024a). The reaction to stimuli refers to the resprouting of new shoots following branch pruning. This process requires input from the pruning locations on branches in the QSM.

The decision of where to prune is the game's most impactful operation for tree growth. Technically, marking any branch node as a pruning point is possible. However, when the game offers such enormous decision-making freedom, it would be too much effort for a human player to click on individual branches to prune them every turn. Meanwhile, a machine player would consume too much computational capacity to test different input combinations. To tackle this problem, four typical pruning strategies in gardening practice were predefined in the game: thinning, raising, reduction, and topping (Clark and Matheny, 2010; Speak and Salbitano, 2023). Each pruning strategy allows users to enter further parameters to specify the operation. The thinning strategy prunes branches whose minimum distance to other branches is less than a given threshold (see Figure 4a). The raising strategy prunes branches below a given height above the crown start (see Figure 4b). The reduction strategy prunes branches from the furthest stretch of a given direction (from west, east, north, south, or top) until a given distance (see Figure 4c). The topping strategy prunes branches within a given depth of every branch's fine end (see Figure 4d). Besides these four pruning operations, the game allows the player to take no action in a round or manually end the episode in the operation phase.

After the pruning operation, a set of random starting positions of cylinders were selected as buds that produce new shoots. These positions kept a minimum of 4 cylinders between each other. Based on these positions, the shooting angle, length, and radius are given within defined domains.

Regarding the generic growth component, we performed an L-system-like growth simulation (Prusinkiewicz and Lindenmayer, 1996). Each cylinder in QSM is considered a node in an L-system, which can have independent elongation in length, increment in radius, and death when experiencing competition (Hemmerling *et al.*, 2008). Such growth simulations have been well explored in functional structural plant models (FSPMs), ranging in different




**Note(s):** Each pruning operation allows additional parameters to adjust the effects, such as distance, depth, and direction

**Source(s):** Authors' own work

**Figure 4.** Four predefined pruning strategies for the players to choose from

scales and integrating various physiological perspectives (Louarn and Song, 2020). FSPMs can simulate the tree growth closer to reality by inputting detailed environmental parameters like lighting, soil water content, etc. However, these can increase the computational load to train the decision-making model. So, in the first version, the generic growth consisted of only three steps: (1) extending the shoots with new cylinders on their tops; (2) increasing cylinder radius in a reverse proportion to their branch order, where the annual *DBH* increment was in reference to Dervishi *et al.* (2022), (3) every branch that grows close to another branch of a larger or similar size has a chance of being deleted. This chance of death due to competition increases with the branching order, so the main trunk becomes more competitive than the sub-branches. Following these three steps, the game produces the future branch state in QSM after growth.

Based on this QSM, the new leaf state in LAD voxel values is estimated (Shu *et al.*, 2024a). The voxel size was 0.8 m on each side to align with the LAD estimation model. The total voxel space consists of $20 \times 20 \times 26$ voxels. So, the whole voxel space has a dimension of over $16 \times 16 \times 20$ meters. A common urban tree grown in urban areas would not exceed this size. It should be noted that LAD values in the voxels are continuous numbers. To acquire a clearer visualization for human players, a visible threshold was set to 0.05 $m^2/m^3$ to emphasize the major crown geometry (see Figure 3b). These results are updated in the user interface.

*2.1.2 A further simplified version of the tree growth game.* In the second round of simplification, the branch primitives in QSM were completely removed to reduce storage consumption further. The leaf primitives were also simplified from LAD values for each voxel to binary Boolean values. The voxels with true values were solid voxels with leaves, while those with false values were void voxels without leaves. Besides, the initial leaf state and target in voxels were fixed to only one configuration for all the episodes.

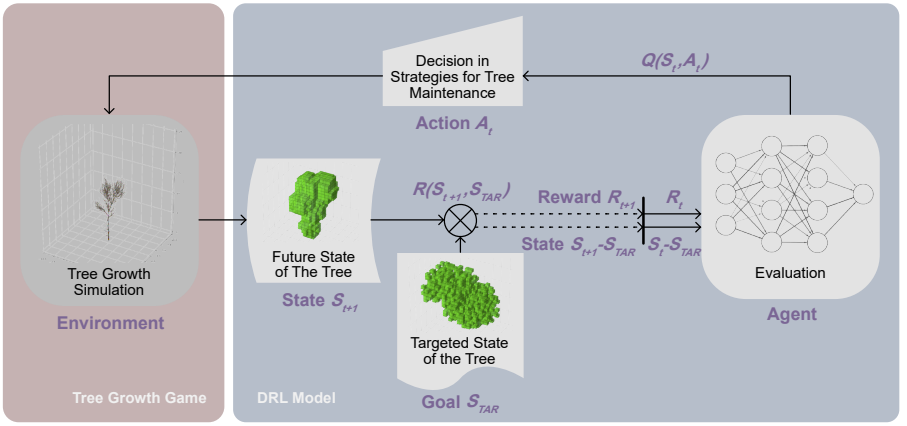
The pruning actions remained the same. But instead of affecting branches, these actions directly delete the solid voxels based on the given depth of the voxel number or a deleting rate (see appendix table A1). Without regrowth of the shoot after pruning, the cause and outcome between a prune operation and the deletion in solid voxels were expected to be clearer. The generic growth of the crown was redrafted accordingly: a solid voxel would expand one voxel wide upward and toward the 4 horizontal directions.

Finally, the game determines if it goes for another iteration or ends this episode depending on trigger conditions. Typical trigger conditions are reaching certain similarities to the target, reaching a maximum round number limit, or receiving a stop command from the player.

### *2.2 Decision-making in the tree growth game with DRL*

As shown in Figure 5, in reinforcement learning, a game consists of (1) an agent, representing the player, (2) the state of the agent (noted as $S_t$, where $t$ is the number of game rounds), (3) an environment where the agent operates, (4) actions available to the agent (noted as $A_t$), and (5) rewards or penalties given to the player for completing or failing certain tasks (noted as $R_t$). The final decision is made by selecting the action with the highest Q-value (noted as $Q(S_t, A_t)$), corresponding to the current state $S_t$ and every possible action $A_t$. The tree growth simulation serves as the environment in this system that provides feedback on the subsequent state of the tree based on the current state and the input action. Instead of a human player, a neural network is the agent that estimates the Q-value using the Bellman equation (Watkins and Dayan, 1992) for every possible state-action pair. These estimates are based on the previously received state and reward as input.

The choice of a specific DRL neural network architecture for training largely depends on the complexity and diversity of the input state and action space. Traditional Deep Q-learning network (DQN) algorithms (Mnih *et al.*, 2015) are well-suited for problems involving discrete action spaces, for example, moving two paddles either up or down a fixed distance per frame in a game of Pong (Diallo *et al.*, 2017). Deep Deterministic Policy Gradient (DDPG) algorithms (Lillicrap *et al.*, 2019) excel in continuous action spaces, for example, releasing the stones with




**Source(s):** Authors' own work

**Figure 5.** Structure of the DRL model in decision-making

a certain direction and velocity in a curling game (Lee *et al.*, 2018). However, many real-world applications require operating in a combined discrete-continuous action space. In optimizing tree pruning strategies, decisions must encompass both the selection of a pruning type (discrete) and the specification of intensity parameters, such as distance or depth (continuous). The Parameterized Deep Q-Network (P-DQN) algorithm (Xiong *et al.*, 2018) extends the capabilities of DQN and DDPG by handling such hybrid discrete-continuous action spaces. The core idea behind P-DQN is to use a deep neural network to estimate the Q-values for discrete actions while simultaneously learning the parameters for continuous actions of each discrete action. The system's objective is for the agent (P-DQN) to learn an optimal policy $\pi$ to sample from action $A_t$ from the actions defined in expression (1). This set consists of tuples where each tuple includes a discrete action $k$ selected from a set of action $K$ and a continuous parameter $x_k$ associated with each $k$. A comprehensive overview of the pruning options $K$ and their corresponding confidence bounds $\mathcal{M}_k$ were listed in appendix table A1.

$$A = \{(k, x_k) | \, x_k \in \mathcal{M}_k, \forall k \in K\} \tag{1}$$

Based on this framework of the DRL model, further details should be set for 1) the architecture of P-DQN (see "Agent" in Figure 5), and 2) the reward function (see "$R(S_{t+1}, S_{TAR})$)" in Figure 5).

The architecture of P-DQN consists of two sub-networks: (1) a dual network to output Q-values for choosing the discrete action $k$ using a Dueling Deep Q-Network (DDQN) (Wang *et al.*, 2016), and (2) a continuous action parameter network that predicts the parameters for each action $x_k$. Two critical phases in training the P-DQN are exploration and exploitation. During exploration, the agent tries out new actions to observe their effects. This phase is essential for learning the accurate responses of the environment. During exploitation, the agent uses prior knowledge from policy $\pi$ to optimize between actions that may further maximize the reward. To balance these two phases, the P-DQN algorithm employs a strategy named epsilon-greedy. If the agent selects a random action with probability $\varepsilon$ in exploration, the action with the highest Q-value will have probability $1 - \varepsilon$ in exploitation. This ensures the agent does not get stuck in suboptimal policies and continues searching for improvements. In training the P-DQN, prior experiences must be stored in a replay buffer, including all past states, actions, rewards, and next states. This consumes a high storage in RAM. During training, mini-batches of these prior data are sampled from the replay buffer to calculate the Q-value. Then, the

Q-value is used to update a least-squares loss function, one of the most common loss functions in DQN. The loss eventually drives the P-DQN network to "learn" a better policy $\pi$.

The general principle for the reward function is to reflect how close the tree state $S_t$ is to the target state $S_{TAR}$. Intersection over Union (IoU) index (Li *et al*., 2021) was used to quantify this similarity. It describes the percentage of common voxels from the two voxel sets $S_t$ and $S_{TAR}$ among the union of them. In order that the agent could see these two sets of voxel states, the state that we used for training the P-DQN at round $t$ is calculated as $S_t - S_{TAR}$.

Using all the settings introduced above, the DRL model and the tree growth game were run on a virtual machine equipped with eight virtual CPUs, a GPU t6435.nvidia-v100.1, and 32 GB RAM. Due to limitations in computational resources, the batch size and network dimensions were optimized to reduce RAM occupation (see section 2.3). Despite these optimizations, computational constraints also affected the number of training episodes that could be stably executed. Given the insufficient number of training episodes in the initial tree growth simulation (see section 4.1), we developed a further simplified version of the tree growth game (see section 2.1.2) to gain a deeper understanding of the novel workflow's effects and limitations (see Section 4.2).

*2.3 Further implementation details*

An upper limit for round numbers was set for a single episode. In the first trained game version, the episode will be terminated after 20 rounds of decision-making and tree growth, and the tree states are reset. The batch size was set to only 64 due to RAM constraints. The dimensions of the critic and actor hidden layers were tuned to $256 \times 128 \times 64$ to balance overfitting and underfitting. Initially, random actions were selected for the first 64 batches to establish a baseline for Q-value estimation. Afterward, the P-DQN network was trained using the prior actions and began choosing actions, although random actions were still selected periodically. The discount factor gamma was set to 0.99 to calculate the Q-value using the Bellman equation. This parameter configuration biases the agent towards the exploitation phase, favoring actions determined by the P-DQN network while allowing for some degree of exploration through random action selection. Learning rates was tuned from 0.001 (Xiong *et al*., 2018) to $1e{-}4$ to ensure stable updates. In addition, an unrealistic parameter in the action could chop off the whole tree. In this case, if the total cylinder numbers of the QSM state went below 20, this episode would terminate immediately with a negative reward of $-1$. Finally, the IoU scores were calculated only for voxels with LAD values larger than 0.05 $m^2/m^3$. A penalty of $-0.2$ points was applied to every round later than 10 to encourage the model to approach the target in earlier rounds. The effectiveness of the reward and penalty system is discussed in section 4.2.

In the further simplified game version with binary voxels, the maximum round limit for each episode was increased to 30. The batch size was increased to 512 to lower RAM usage when running the simplified game. The dimensions for critic and actor hidden layers were accordingly upgraded to $256 \times 128 \times 64 \times 32$, and convolutional neural network (CNN) was used as feature extractors in the hidden layers. The discount factor gamma was still set to 0.99 encountering any issues. The learning rate was further tuned to $1e{-}5$ to reduce loss fluctuations and stabilize training. An upper limit for episodes was set to 2000, which was not reached (see section 3.2). For bad actions that deleted all the solid voxels, the penalty was increased to $-100$ to be comparable with the cumulative rewards within 30 rounds. The state feeding to the P-DQN network used in the simplified game was exactly the binary voxels of the current leaf areas.

During training, the agent stores experience tuples—consisting of the current state, selected action, received reward, and next state—into a replay memory implemented as a queue. This memory retains up to 1e6 tuples, discarding the oldest entries as new data is added. To improve learning stability, the agent updates its parameters twice per step by randomly sampling from stored experiences. This practice is widely accepted for mitigating the risk of overfitting to specific experiences sequences (Mnih *et al*., 2015). The source code for the game and the DRL model, including these detailed settings, is available on our GitHub page (Shu and Boey, 2024).

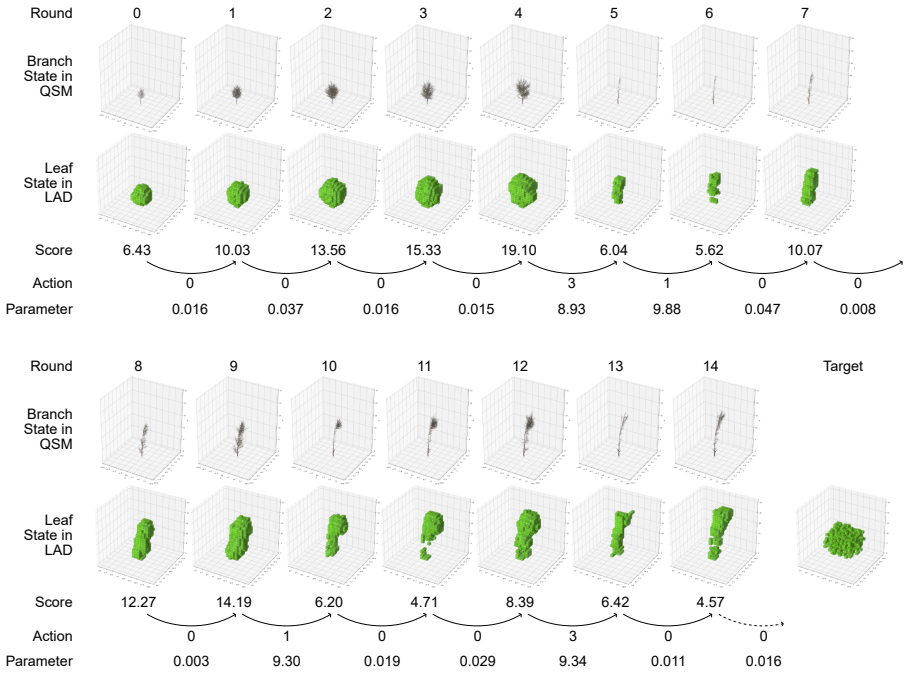
## 3. Results

### *3.1 The model trained on the first version of the tree growth game*

The training process on the first game version managed to reach 60 episodes. For every episode, their total rounds, consumed time, and the frequency of chosen actions are shown in Appendix Figure A1. The first 7 episodes used random actions. From the 8th episode onwards, the P-DQN network took effects to make decisions on actions. With these decisions, more than half of the episodes lasted longer than 15 rounds (see appendix Figure A1a), which means the parameters of those actions were, at least, not chopping the whole tree off. This was also indirectly proven by the time consumption (see appendix Figure A1b). Longer time in the growth simulation at later episodes indicated the tree had grown larger with more cylinders.

The most chosen actions by the P-DQN network had shifted from Raising (action 1) to Thinning (action 0) and Reduction from the south (action 3) (see appendix Figure A1c). Among these 3 most used actions (see appendix Figure A1d), the thinning was carried out very gently in most of the cases with a minimum distance threshold for branches less than 5 mm. On the contrary, raising the crown was carried out much more intensively with a raised height of mostly 9–10 meters. The reduction from the south was carried with both short- and long-distance depth.

Figure 6 illustrates the changes in the branches and leaves of the tree in episode 55. Round 4 in this episode has won the highest maximum reward in all the episodes. Each round was also noted with its corresponding scores and decisions in action. Most thinning decisions (action 0) were with a small parameter. They generally encouraged the free growth of the crown, which won mostly an increasing score. However, the two raising decisions (action 1 at round 5 and 9) and the two reductions from the south (action 3 at round 4 and 12) cut too many branches, reducing the IoU score significantly.




**Source(s):** Authors' own work

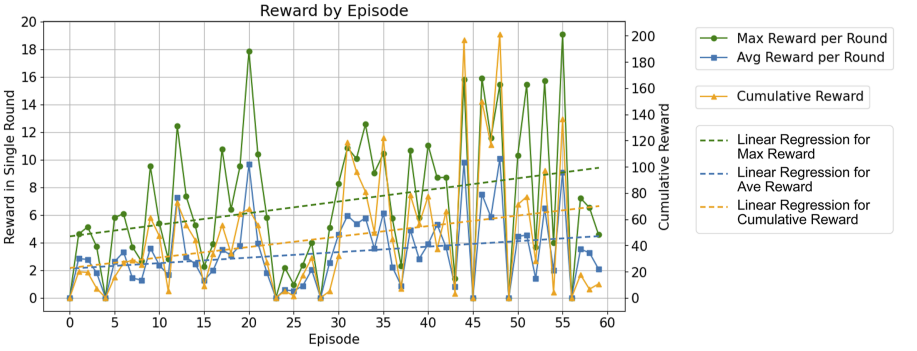
**Figure 6.** Rendered records of the tree growth game at Episode 55 with the maximum rewards at round 4

In an overall trend (see Figure 7), the maximum reward in each episode, the average reward, and the cumulative reward all climb with the increasing training episode numbers. The cumulative reward and maximum reward climb faster than the average reward. Within the 60 episodes, a stable pruning strategy had not been achieved. Therefore, we could not see the further development of these gaining rewards.

### *3.2 The model trained on the further simplified binary game*

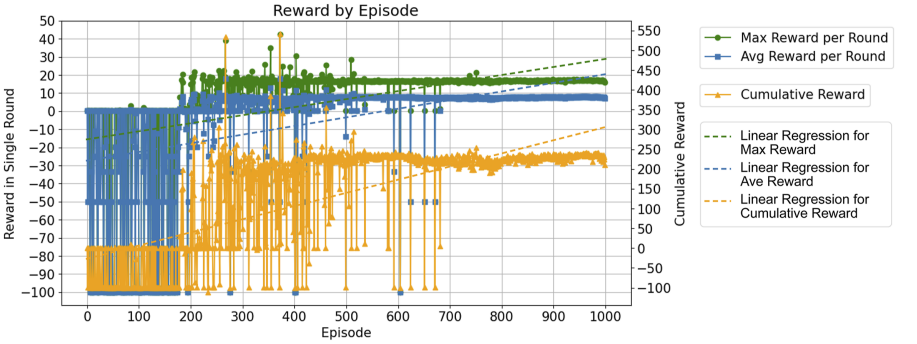
In the training with the binary tree growth game, 1,000 episodes were easily reached. The first 175 episodes used random actions, while the P-DQN network decided the actions in later episodes. A stabilized pruning strategy was achieved after 700 episodes (see Figure 8). Statistics regarding time consumption and frequency of actions are shown in appendix figure A2. Most of the episodes with 30 rounds were finished within 10 s (see appendix figure A2b). The most chosen actions changed from reduction from the south to topping at around 200 to 300 episodes. Afterward, the thinning dominated the pruning decisions (see appendix figure A2c). Among these three most used actions, the thinning (action 0) was conducted mostly gently and some with middle intensity. The reduction from the south (action 3) and topping (action 7) mainly used a short distance or depth (see appendix figure A2d).

The average and maximum reward per round and cumulative reward in the episode are shown in Figure 8. The P-DQN network rapidly learned to gain positive rewards after the 175




**Source(s):** Authors' own work

**Figure 7.** The trend of the gained reward using linear regression




**Source(s):** Authors' own work

**Figure 8.** The trend of the gained reward in the simplified tree growth game with binary voxels

episodes with random actions. The reward was stabilized at 10 points per round on average at the end of the training. It found the "best" pruning strategy to win the highest reward in the binary tree growth game. The visualization of this process in episode 800 is illustrated in appendix figure A3. From this record, it was recognized that this strategy was almost a free expansion of the crown. Even when the solid voxels occupied almost the whole voxel space, the reward dominated by the IoU score was still promising. So, it has made the decision-making model too "lazy" to use other actions to approach a more precise crown geometry of the target.

## 4. Discussion

### *4.1 Evaluation of the trained model in the first game version*

Based on the limited training episodes in the first simplified game, we can preliminarily conclude that the P-DQN network has gradually learned to avoid the penalty reward −1. It also means it has learned not to terminate the tree growth game by reducing the cylinder number below 20. This is explained by the climbing scores and increasing simulation time in the results.

However, for the specific action selection, it doesn't seem like this network recognized the logic between the action, its parameters, the change of the tree states, and the reward. If a human player was handling the tree state in round 4 of episode 55, a better decision might be a reduction from the top to stop the growth in height while encouraging a horizontal expansion of the crown. But the trained P-DQN network was satisfied with cutting almost all the side branches away to win a smaller score. It repeated a similar choice later at round 12. It failed to explore other actions and parameters to win higher rewards.

There could be multiple reasons for such behaviors of the P-DQN network: (1) The major deficiency was the lack of episodes for exploring different combinations of action and parameters with random settings, especially for tree states with a larger crown. Random parameters had a high chance of killing the tree and terminating the game when the tree was small. Meanwhile, the high reward can only happen later in the game when the crown has grown large. Such tree states with large crowns only occurred when all the random actions were, by chance, preserving the tree growth. Therefore, a high number of experiments were required for the network to learn the effects of different actions on large trees. In addition, the frequency of actions and parameters shown in Appendix Figure A1 were unbalanced. These distributions indicate the insufficient diverse tree states and action combinations explored through the training process. Apparently, current tries did not make the network "experienced" enough with the huge action space under different tree states. (2) The clarity of rewards and penalties is another key factor in determining training performance. The IoU index used in the experiments gave a positive reward based on the percentage of an overlap of the tree state with the target voxels. Even when the state after the action overlapped less than the previous state, the reward was still positive but smaller. This may lead to confusion in identifying the true good actions. (3) The balance between exploitation and exploration by the gamma value also affected the results. It determined if the model relied more on familiar paths or would take risks exploring new actions. Setting the gamma to 0.99 may have strengthened the P-DQN network in taking actions and parameters that were accidentally tried in the random settings with smaller rewards. It became too conservative to explore other new actions and parameters.

The detailed design choices regarding rewards, penalties for suboptimal actions, and round limits above were determined based on the authors' expertise, complemented by iterative adjustments through multiple test runs. These revisions aimed to balance the learning process and ensure stable training dynamics. However, the current reward system may not represent its most optimized form. Further refinements and potential optimization strategies are discussed in Section 4.2.

### *4.2 Reflections on the results from the further simplified versions of games*

Two valuable abilities of the P-DQN network have been addressed through the binary version of games: (1) it can handle a large hybrid action space consisting of discrete operation types and a parameter within a continuous domain to describe the intensity of the operation. (2) it can effectively

get the proper operations to avoid penalties and gain relatively high rewards. At the same time, the following two obstacles were also seen in this work before it could be applied to the industry.

Due to computational constraints in this study, training a P-DQN network on a physiologically based tree growth simulation was not feasible. Instead, our experiments relied on a simplified tree growth simulation using binary voxels. While this approach significantly reduced computational demands, it does not fully capture the complexity of real tree growth dynamics. The binary voxel representation lacks physiological accuracy, meaning the simulated growth outcomes may not directly reflect real-world tree responses to pruning.

The objective of achieving a target leaf area remains ambiguous compared to well-defined tasks in other reinforcement learning applications, such as catching a ping-pong ball (Diallo *et al*., 2017). This ambiguity makes it challenging to establish a precise reward indexing system that effectively guides tree growth toward a desired crown geometry. As a result, the trained model may achieve high rewards through unintended or suboptimal strategies rather than following biologically meaningful growth patterns. To improve reward optimization, future studies could explore methodologies such as curriculum learning (Portelas *et al*., 2020), where the learning process is structured in a way that gradually increases the difficulty of tasks. Additionally, reward-shaping techniques (Viswanadhapalli *et al*., 2024) could be implemented to modify rewards that better align with the long-term objective of crown formation.

### *4.3 Visions of this study*

Future developments in the hardware will solve the first obstacle regarding limitations in computational power. The second obstacle requires optimizing the reward system, the state description, or the network structure. We open-source the tree growth game to welcome researchers and other experts in reinforcement learning to test other parameters, reward indexes, and network structures for training a DRL model in playing the tree growth game.

Nevertheless, the configured P-DQN network was proven a feasible technical route in decision-making for tree pruning. It served as a starting point for even adding other tree-management decisions in the future.

Moreover, the tree growth games have proven flexible. They can be either simplified to reduce the computational load or deepened in certain aspects to address new features and boundary conditions, such as root distribution, water content, etc. In this way, the proposed framework combining a growth game and a DRL network can, in the future, also integrate new physiological models for simulating tree growth and even enriched data formats in describing tree states.

Finally, transitioning from single-tree decision-making to urban-scale green infrastructure management would necessitate more efficient algorithms or cloud-based distributed computing solutions (Popović *et al*., 2018; Rashid *et al*., 2018). As this transition also involves a substantial increase in data volume, this accumulation may paradoxically reduce computational costs and requirements. By leveraging empirical and tacit knowledge such as knowledge graphs (Wu, 2024) or advanced models like DeepSeek (DeepSeek-AI *et al.*, 2024), patterns and relationships within the data can be systematically captured, enabling more efficient decision-making. Similar approaches may facilitate scalability for large-scale green infrastructure management in the future.

At that stage, real-time or simulated pruning decisions can be integrated into digital twins, enabling urban planners to monitor and optimize the long-term impacts of tree growth on ecosystem services using voxel-based representations. An arborist could receive real-time updates on target crown adjustments in voxel representations using a tablet or AR glasses. The decision-support tool would provide a visual guide for shaping the tree and estimate the number of years required to achieve the target geometry. Additionally, it could generate an optimized pruning schedule, specifying which branches should be removed in each phase, the appropriate tools required, and the estimated labor investment needed for execution. Such precise information and intuitive visualization enhance data-driven urban forestry

management and foster public engagement in green infrastructure planning within broader smart city frameworks.

## 5. Conclusion

In a traditional workflow of designing and managing urban trees, designers drafted a fixed tree size on paper before the implementation. In contrast, the tree management specialists work hands-on for a longer period. Decisions for pruning branches were made without clear design intentions regarding the crown shapes. This workflow cannot address the multifunctional use of urban trees, offering larger shading areas to confront global warming. Therefore, we proposed a workflow where designers set targets for leaf areas based on different boundary conditions. The tree management specialists can be advised by a decision-support algorithm in tree management, especially for pruning, to guide the tree growth toward the design target.

Existing tools support the proposed workflow, including digital tree twins, LiDAR scanning, and QSM extraction. Tree growth simulation is feasible with FSPMs and resprouting predictions, while voxels serve as geometric primitives for defining leaf area targets. What is still missing is a decision-support mechanism that guides tree growth toward the design target. Therefore, this paper explored the feasibility of DRL in decision-making for tree management to complete this workflow.

A novel framework was developed to train a DRL network for this purpose. It consists of a tree growth game and a decision-making model. The tree growth game served as a simulation environment. It predicted the future state of a tree based on its reaction to stimuli and related growth simulations. This determined mostly a rewarding score. A P-DQN network was the decision-making agent. It received the states of trees and their corresponding rewards largely determined by the IoU index from the game. Based on prior experiments, Q-values were evaluated to tell which actions to take under which states of the tree to win a higher reward. The action space was restricted to 4 classical pruning practices: thinning, raising, reduction, and topping. Each pruning strategy can be described with an additional parameter to specify their intensities, such as depth and distances. By playing the tree growth game iteratively, accumulated experiences were expected to make the P-DQN network attain a smart pruning strategy.

With all the RAM space we had for storing the training data, the first trained model only completed 60 episodes in the game. Diverse tree states and action combinations were insufficiently explored. Within these limits, the increasing trend of the reward was clear. However, when looking at specific actions taken in different states, the reward could become even higher with “smarter” decisions. However, the P-DQN network cannot fully explore these possibilities and see their benefits. To get over this obstacle, the tree growth game was further simplified. Binary voxels were used to describe the occupancy of the leaves. Solid leaf voxels would expand to surrounding spaces in each round and could be deleted by pruning actions. There was no longer a complex tree growth simulation. With this binary version of the game, we could train the P-DQN network over 1,000 episodes and reach a stable pruning strategy. This model effectively learned to avoid the penalty of chopping off the tree and found a good “trick” in getting a high reward: allowing the canopy to grow freely and occupy the whole voxel space. Unfortunately, this way of getting a high reward was still not the purpose of developing this model.

With our experiments, we could prove that the P-DQN network has two great abilities in decision-making for tree management strategies: (1) making decisions in a hybrid action space that decides both discrete operation types and an additional continuous parameter to describe their intensities. (2) its effectiveness in ruling out actions that caused penalties and could increase the reward little by little, even in such a large decision space and complex environment. Except for its strength, we have also found two obstacles before this model could be applied to the industry: (1) lack of computational resources to train this decision support model with enough episodes in a close-to-real tree growth simulation environment. This can

probably be solved by hardware development, more efficient algorithms, or distributed computing solutions. The future accumulation of urban tree data may also paradoxically reduce computational costs and requirements by leveraging empirical and tacit knowledge. (2) Another tricky task is to create a clear reward index to achieve the desired model outcome. Otherwise, the P-DQN network learned to win a good score but did not get the tree crowns close to the design target. This problem may be solvable by curriculum learning or reward-shaping techniques. Testing these alternatives requires interdisciplinary cooperation. Therefore, we released our source code for open access to encourage further tests from other researchers and experts.

Nevertheless, this work has ascertained a technical route to realize the proposed novel workflow, where the computational approach is used in complex decision-making regarding pruning positions on branches to guide the tree growth approaching a design goal. Beyond its technical contributions, the voxel-based approach for setting design targets provides an intuitive representation of urban trees within digital twins, facilitating collaboration among urban planners, arborists, municipal authorities, and even the public. This collaborative approach is crucial for optimizing long-term green infrastructure management while enhancing various ecosystem services, including urban cooling, stormwater regulation, and air quality improvement. In this regard, this study lays the foundation for the multifunctional use of urban trees, aligning with global challenges and contributing to developing resilient and climate-adaptive urban environments.

## References

Abegg, M., Kükenbrink, D., Zell, J., Schaepman, M.E. and Morsdorf, F. (2017), "Terrestrial laser scanning for forest inventories—tree diameter distribution and scanner location impact on occlusion", *Forests*, Vol. 8 No. 6, p. 184, 6, doi: 10.3390/f8060184.

Bedker, P., O'Brien, J. and Mielke, M. (2012), "How to prune trees", *USDA, Forest Service, State and Private Forestry, Northeastern Area, 11 Campus Blvd., Ste 200 Newtown Square, PA 19073, FR-01-95*, available at: https://www.fs.usda.gov/research/treesearch/12602

Bittencourt, J.C.N., Costa, D.G., Portugal, P. and Vasques, F. (2024), "A survey on adaptive smart urban systems", *IEEE Access*, Vol. 12, pp. 102826-102850, doi: 10.1109/ACCESS.2024.3433381.

Brandt, M., Chave, J., Li, S., Fensholt, R., Ciais, P., Wigneron, J.-P., Gieseke, F., Saatchi, S., Tucker, C.J. and Igel, C. (2025), "High-resolution sensors and deep learning models for tree resource monitoring", *Nature Reviews Electrical Engineering*, Vol. 2 No. 1, pp. 13-26, doi: 10.1038/s44287-024-00116-8.

Chau, W.Y., Wang, Y.-H., Chiu, S.W., Tan, P.S., Leung, M.L., Lui, H.L., Wu, J., Lau, Y.M., Liu, K.-F. and Hau, B.C.H. (2023), "Monitoring of tree tilt motion using lorawan-based wireless tree sensing system (IoTT) during super typhoon Mangkhut", *Agricultural and Forest Meteorology*, Vol. 329, 109282, doi: 10.1016/j.agrformet.2022.109282.

Clark, J. and Matheny, N. (2010), "The research foundation to tree pruning: a review of the literature", *Arboriculture and Urban Forestry*, Vol. 36 No. 3, pp. 110-120, doi: 10.48044/jauf.2010.015.

Dan, L., Yong, P. and CaiRong, Y. (2012), "A review of TLS application in forest parameters retrieving", *World Forestry Research*, Vol. 25 No. 6, pp. 34-39.

Dervishi, V., Poschenrieder, W., Rötzer, T., Moser-Reischl, A. and Pretzsch, H. (2022), "Effects of climate and drought on stem diameter growth of urban tree species", *Forests*, Vol. 13 No. 5, p. 641, doi: 10.3390/f13050641.

Diallo, E.A.O., Sugiyama, A. and Sugawara, T. (2017), "Learning to coordinate with deep reinforcement learning in doubles pong game", *2017 16th IEEE International Conference on Machine Learning and Applications (ICMLA)*, pp. 14-19, doi: 10.1109/ICMLA.2017.0-184.

Dowtin, A.L., Cregg, B.C., Nowak, D.J. and Levia, D.F. (2023), "Towards optimized runoff reduction by urban tree cover: a review of key physical tree traits, site conditions, and management

strategies", *Landscape and Urban Planning*, Vol. 239, 104849, doi: 10.1016/j.landurbplan.2023.104849.

Du, S., Lindenbergh, R., Ledoux, H., Stoter, J. and Nan, L. (2019), "AdTree: accurate, detailed, and automatic modelling of laser-scanned trees", *Remote Sensing*, Vol. 11 No. 18, doi: 10.3390/rs11182074.

DeepSeek-AI, Liu, A., Feng, B., Xue, B., Wang, B., Wu, B., Lu, C., Zhao, C., Deng, C., Zhang, C., Ruan, C., Dai, D. (2024), *DeepSeek-V3 Technical Report (No. arXiv:2412.19437)*, arXiv, doi: 10.48550/arXiv.2412.19437.

Fan, G., Nan, L., Dong, Y., Su, X. and Chen, F. (2020), "AdQSM: a new method for estimating above-ground biomass from tls point clouds", *Remote Sensing*, Vol. 12 No. 18, doi: 10.3390/rs12183089.

Francis, J., Disney, M. and Law, S. (2023), "Monitoring canopy quality and improving equitable outcomes of urban tree planting using LiDAR and machine learning", *Urban Forestry and Urban Greening*, Vol. 89, 128115, doi: 10.1016/j.ufug.2023.128115.

Fu, M.C. (2016), "AlphaGo and monte carlo tree search: the simulation optimization perspective", 2016 Winter Simulation Conference (WSC), 11-14 December 2016 at Washington, DC, IEEE, pp. 659-670, doi: 10.1109/WSC.2016.7822130.

Grylls, T. and Van Reeuwijk, M. (2021), "Tree model with drag, transpiration, shading and deposition: identification of cooling regimes and large-eddy simulation", *Agricultural and Forest Meteorology*, pp. 298-299, doi: 10.1016/j.agrformet.2020.108288.

Helmes, K.L. and Stockbridge, R.H. (2011), "Thinning and harvesting in stochastic forest models", *Journal of Economic Dynamics and Control*, Vol. 35 No. 1, pp. 25-39, doi: 10.1016/j.jedc.2010.10.007.

Hemmerling, R., Kniemeyer, O., Lanwert, D., Kurth, W. and Buck-Sorlin, G. (2008), "The rule-based language XL and the modelling environment GroIMP illustrated with simulated tree competition", *Functional Plant Biology*, Vol. 35 No. 10, pp. 739-750, doi: 10.1071/FP08052.

Holmgren, P. and Thuresson, T. (1998), "Satellite remote sensing for forestry planning—a review", *Scandinavian Journal of Forest Research*, Vol. 13 Nos 1-4, pp. 90-110, doi: 10.1080/02827589809382966.

Kohek, Š., Guid, N., Tojnko, S., Unuk, T. and Kolmanič, S. (2015), "EduAPPLE: interactive teaching tool for apple tree crown formation", *HortTechnology*, Vol. 25 No. 2, pp. 238-246, doi: 10.21273/HORTTECH.25.2.238.

Krayenhoff, E.S., Christen, A., Martilli, A. and Oke, T.R. (2014), "A multi-layer radiation model for urban neighbourhoods with trees", *Boundary-Layer Meteorology*, Vol. 151 No. 1, pp. 139-178, doi: 10.1007/s10546-013-9883-1.

Lee, K., Kim, S.-A., Choi, J. and Lee, S.-W. (2018), "Deep reinforcement learning in continuous action spaces: a case study in the game of simulated curling", *Proceedings of the 35th International Conference on Machine Learning*, pp. 2937-2946, available at: https://proceedings.mlr.press/v80/lee18b.html

Li, J., Dai, H., Shao, L. and Ding, Y. (2021), "From voxel to point: IoU-guided 3D object detection for point cloud with voxel-to-point decoder", *Proceedings of the 29th ACM International Conference on Multimedia*, pp. 4622-4631, doi: 10.1145/3474085.3475314.

Li, C., Zheng, P., Yin, Y., Wang, B. and Wang, L. (2023), "Deep reinforcement learning in smart manufacturing: a review and prospects", *CIRP Journal of Manufacturing Science and Technology*, Vol. 40, pp. 75-101, doi: 10.1016/j.cirpj.2022.11.003.

Lillicrap, T.P., Hunt, J.J., Pritzel, A., Heess, N., Erez, T., Tassa, Y., Silver, D. and Wierstra, D. (2019), *Continuous Control with Deep Reinforcement Learning* (No. arXiv:1509.02971), arXiv, doi: 10.48550/arXiv.1509.02971.

Louarn, G. and Song, Y. (2020), "Two decades of functional–structural plant modelling: now addressing fundamental questions in systems biology and predictive ecology", *Annals of Botany*, Vol. 126 No. 4, pp. 501-509, doi: 10.1093/aob/mcaa143.

Ludwig, F., Hensel, M., Rötzer, T., Ahmeti, A., Chen, X., Erdal, H.I., Reischel, A., Shu, Q., Tyc, J.M. and Yazdi, H. (2024), "Digital workflow for novel urban green system design derived from a historical role model", *Journal of Digital Landscape Architecture*, Vol. 2024 No. 9, pp. 333-345, doi: 10.14627/537752030.

Mnih, V., Kavukcuoglu, K., Silver, D., Graves, A., Antonoglou, I., Wierstra, D. and Riedmiller, M. (2013), *Playing Atari with Deep Reinforcement Learning* (No. arXiv:1312.5602), arXiv, doi: 10.48550/arXiv.1312.5602.

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A.A., Veness, J., Bellemare, M.G., Graves, A., Riedmiller, M., Fidjeland, A.K., Ostrovski, G., Petersen, S., Beattie, C., Sadik, A., Antonoglou, I., King, H., Kumaran, D., Wierstra, D., Legg, S. and Hassabis, D. (2015), "Human-level control through deep reinforcement learning", *Nature*, Vol. 518 No. 7540, pp. 529-533, doi: 10.1038/nature14236.

Nitoslawski, S.A., Galle, N.J., Van Den Bosch, C.K. and Steenberg, J.W.N. (2019), "Smarter ecosystems for smarter cities? A review of trends, technologies, and turning points for smart urban forestry", *Sustainable Cities and Society*, Vol. 51, 101770, doi: 10.1016/j.scs.2019.101770.

Oshio, H., Kiyono, T. and Asawa, T. (2021), "Numerical simulation of the nocturnal cooling effect of urban trees considering the leaf area density distribution", *Urban Forestry and Urban Greening*, Vol. 66, 127391, doi: 10.1016/j.ufug.2021.127391.

Palme, M., La Rosa, D., Privitera, R. and Chiesa, G. (2019), "Evaluating the potential energy savings of an urban green infrastructure through environmental simulation", *Proceedings of the 16th IBPSA Conference*, Rome, Italy, Sept 2-4, 2019, pp. 3524-3530, doi: 10.26868/25222708.2019.210698.

Pan, J. and Jakubiec, J. (2022), "Simulating the impact of deciduous trees on energy, daylight, and visual comfort: impact analysis and a practical framework for implementation", *Proceedings of eSim Building Simulation Conference 2022: 12th Conference of IBPSA-Canada*, June 22-23, 2022 at Ottawa, Canada.

Popović, N.D., Popović, D.S. and Seskar, I. (2018), "A novel cloud-based advanced distribution management system solution", *IEEE Transactions on Industrial Informatics*, Vol. 14 No. 8, pp. 3469-3476, IEEE Transactions on Industrial Informatics, doi: 10.1109/TII.2017.2780060.

Portelas, R., Colas, C., Weng, L., Hofmann, K. and Oudeyer, P.-Y. (2020), *Automatic Curriculum Learning for Deep RL: A Short Survey* (No. arXiv:2003.04664), arXiv, doi: 10.48550/arXiv.2003.04664.

Prusinkiewicz, P. and Lindenmayer, A. (1996), *The Algorithmic Beauty of Plants*, Springer, New York, NY, doi: 10.1007/978-1-4613-8476-2.

Puterman, M.L. (2014), *Markov Decision Processes: Discrete Stochastic Dynamic Programming*, John Wiley & Sons, Hoboken, NJ, ISBN: 978-1-118-62587-3.

Rahman, M.A., Stratopoulos, L.M.F., Moser-Reischl, A., Zölch, T., Häberle, K.-H., Rötzer, T., Pretzsch, H. and Pauleit, S. (2020), "Traits of trees for cooling urban heat islands: a meta-analysis", *Building and Environment*, Vol. 170, 106606, doi: 10.1016/j.buildenv.2019.106606.

Rambhia, M., Volk, R., Rismanchi, B., Winter, S. and Schultmann, F. (2023), "Supporting decision-makers in estimating irrigation demand for urban street trees", *Urban Forestry and Urban Greening*, Vol. 82, 127868, doi: 10.1016/j.ufug.2023.127868.

Rashid, Z.N., Zebari, S.R.M., Sharif, K.H. and Jacksi, K. (2018), "Distributed cloud computing and distributed parallel computing: a review", *2018 International Conference on Advanced Science and Engineering (ICOASE)*, pp. 167-172, doi: 10.1109/ICOASE.2018.8548937.

Raumonen, P., Kaasalainen, M., Åkerblom, M., Kaasalainen, S., Kaartinen, H., Vastaranta, M., Holopainen, M., Disney, M. and Lewis, P. (2013), "Fast automatic precision tree models from terrestrial laser scanner data", *Remote Sensing*, Vol. 5 No. 2, doi: 10.3390/rs5020491.

Shu, Q. and Boey, K.Z. (2024), "QiguanShu/Branch-pruning-game-on-urban-trees: this is one of the research project named Urban Green System 4.0 funded by DFG-DACH (LU2505/2-1). Please find more details in our publication", *GitHub*, available at: https://github.com/QiguanShu/Branch-Pruning-Game-on-Urban-Trees

Shu, Q., Rötzer, T., Detter, A. and Ludwig, F. (2022), "Tree information modeling: a data exchange platform for tree design and management", *Forests*, Vol. 13 No. 11, doi: 10.3390/f13111955.

Shu, Q., Rötzer, T., Yazdi, H., Moser-Reischl, A. and Ludwig, F. (2024a), *Can Leaf Area Density Be Estimated from Quantitative Structure Models of Trees?* (SSRN Scholarly Paper No. 4855810), doi: 10.2139/ssrn.4855810.

Shu, Q., Yazdi, H., Rötzer, T. and Ludwig, F. (2024b), "Predicting resprouting of Platanus × hispanica following branch pruning by means of machine learning", *Frontiers in Plant Science*, Vol. 15, 1297390, doi: 10.3389/fpls.2024.1297390.

Silva, S., Cardoso, T., Barros, P., Ribeiro, H., Carvalho, P. and Rito Lima, S. (2020), "A flexible system for optimising green spaces irrigation", *2020 5th International Conference on Smart and Sustainable Technologies (SpliTech)*, pp. 1-6, doi: 10.23919/SpliTech49282.2020.9243756.

Silver, D., Huang, A., Maddison, C.J., Guez, A., Sifre, L., van den Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M., Dieleman, S., Grewe, D., Nham, J., Kalchbrenner, N., Sutskever, I., Lillicrap, T., Leach, M., Kavukcuoglu, K., Graepel, T. and Hassabis, D. (2016), "Mastering the game of go with deep neural networks and tree search", *Nature*, Vol. 529 No. 7587, pp. 484-489, doi: 10.1038/nature16961.

Speak, A.F. and Salbitano, F. (2023), "The impact of pruning and mortality on urban tree canopy volume", *Urban Forestry and Urban Greening*, Vol. 79, 127810, doi: 10.1016/j.ufug.2022.127810.

Vinyals, O., Ewalds, T., Bartunov, S., Georgiev, P., Vezhnevets, A.S., Yeo, M., Makhzani, A., Küttler, H., Agapiou, J., Schrittwieser, J., Quan, J. (2017), StarCraft II: A New Challenge for Reinforcement Learning (No. arXiv:1708.04782), arXiv, doi: 10.48550/arXiv.1708.04782.

Viswanadhapalli, J.K., Elumalai, V.K.S.S., Shah, S. and Mahajan, D. (2024), "Deep reinforcement learning with reward shaping for tracking control and vibration suppression of flexible link manipulator", *Applied Soft Computing*, Vol. 152, 110756, doi: 10.1016/j.asoc.2023.110756.

Vos, P.E.J., Maiheu, B., Vankerkom, J. and Janssen, S. (2013), "Improving local air quality in cities: to tree or not to tree?", *Environmental Pollution*, Vol. 183, pp. 113-122, doi: 10.1016/j.envpol.2012.10.021.

Wang, Z., Schaul, T., Hessel, M., van Hasselt, H., Lanctot, M. and de Freitas, N. (2016), *Dueling Network Architectures for Deep Reinforcement Learning* (No. arXiv:1511.06581), arXiv, doi: 10.48550/arXiv.1511.06581.

Wang, X., Wang, S., Liang, X., Zhao, D., Huang, J., Xu, X., Dai, B. and Miao, Q. (2024), "Deep reinforcement learning: a survey", *IEEE Transactions on Neural Networks and Learning Systems*, Vol. 35 No. 4, pp. 5064-5078, doi: 10.1109/TNNLS.2022.3207346.

Watkins, C.J.C.H. and Dayan, P. (1992), "Q-learning", *Machine Learning*, Vol. 8 No. 3, pp. 279-292, doi: 10.1007/BF00992698.

Wong, N.H., Tan, C.L., Kolokotsa, D.D. and Takebayashi, H. (2021), "Greenery as a mitigation and adaptation strategy to urban heat", *Nature Reviews Earth and Environment*, Vol. 2 No. 3, pp. 166-181, doi: 10.1038/s43017-020-00129-5.

Wu, Z. (2024), *An Efficient Recommendation Model Based on Knowledge Graph Attention-Assisted Network (KGATAX)* (No. arXiv:2409.15315), arXiv, doi: 10.48550/arXiv.2409.15315.

Xiong, J., Wang, Q., Yang, Z., Sun, P., Han, L., Zheng, Y., Fu, H., Zhang, T., Liu, J. and Liu, H. (2018), *Parametrized Deep Q-Networks Learning: Reinforcement Learning with Discrete-Continuous Hybrid Action Space* (No. arXiv:1810.06394), arXiv, doi: 10.48550/arXiv.1810.06394.

Yazdi, H., Shu, Q. and Ludwig, F. (2023), "A target-driven tree planting and maintenance approach for next generation urban green infrastructure (UGI)", *JoDLA – Journal of Digital Landscape Architecture*, Vols 8-2023, p. 178, doi: 10.14627/537740019.

Yazdi, H., Shu, Q., Rötzer, T., Petzold, F. and Ludwig, F. (2024), "A multilayered urban tree dataset of point clouds, quantitative structure and graph models", *Scientific Data*, Vol. 11 No. 1, 1, doi: 10.1038/s41597-023-02873-x.

Yu, L., Qin, S., Zhang, M., Shen, C., Jiang, T. and Guan, X. (2021), "A review of deep reinforcement learning for smart building energy management", *IEEE Internet of Things Journal*, Vol. 8 No. 15, pp. 12046-12063, doi: 10.1109/JIOT.2021.3078462.

## Appendix

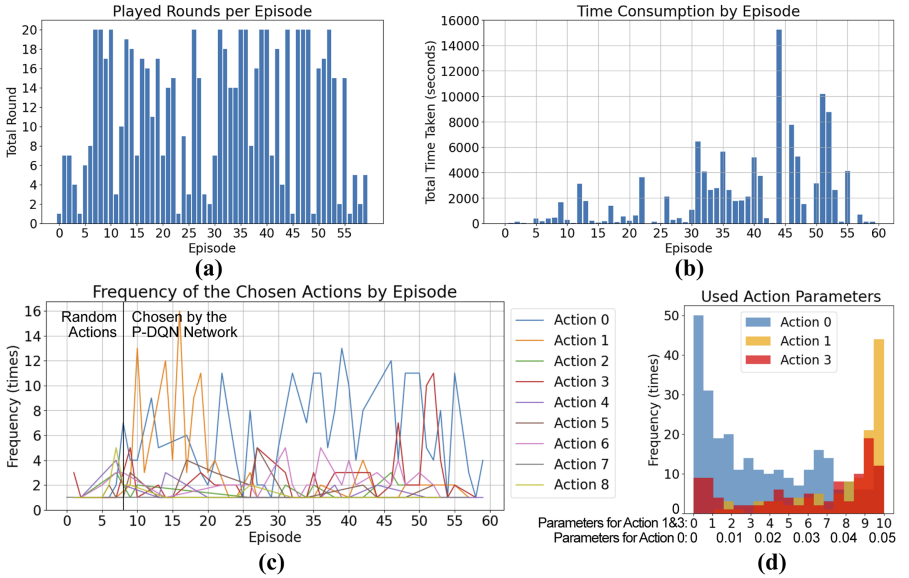
### 1. Definitions for the action space

**Table A1.** Action space for agents to decide in the tree growth game and its simplified binary version

| Action no. | Pruning type $\boldsymbol{K}$ | Range $\mathcal{M}_k$ | Meaning of the parameter | Range (in binary version) $\mathcal{M}_k$ | Meaning of the parameter (in the binary version) |
|---|---|---|---|---|---|
| 0 | Thinning | (float) [0, 0.05] | Branches with a distance to any other branch below this number in meters will be cut | (float) [0, 0.5) | The rate of solid voxels to be deleted |
| 1 | Raising | (float) [0, 10] | Branches within this height in meters from the crown start will be cut | (Integer) [1, 10] | The number of solid voxel layers to be deleted from the bottom |
| 2 | Reduction east | (float) [0, 10] | Branches within this distance, meters from the crown's outreach from the east, will be cut | (Integer) [1, 10] | The number of solid voxel layers to be deleted from the east |
| 3 | Reduction south | (float) [0, 10] | Branches within this distance, meters from the crown's outreach from the south, will be cut | (Integer) [1, 10] | The number of solid voxel layers to be deleted from the south |
| 4 | Reduction west | (float) [0, 10] | Branches within this distance, meters from the crown's outreach from the west, will be cut | (Integer) [1, 10] | The number of solid voxel layers to be deleted from the west |
| 5 | Reduction north | (float) [0, 10] | Branches within this distance, meters from the crown's outreach from the north, will be cut | (Integer) [1, 10] | The number of solid voxel layers to be deleted from the north |
| 6 | Reduction top | (float) [0, 5] | Branches within this distance in meters below the crown's top will be cut | (Integer) [1, 10] | The number of solid voxel layers to be deleted from the top |
| 7 | Topping | (integer) [0, 5] | Cylinders that within this number from an end of a branch will be cut | (float) (0, 5] | Delete voxels whose distance from their center to the mean center of all solid voxels is among the furthest ones within this range in meters |
| 8 | No action (only with a generic growth) | – | Do not conduct any manual pruning | – | Do not conduct any manual pruning |

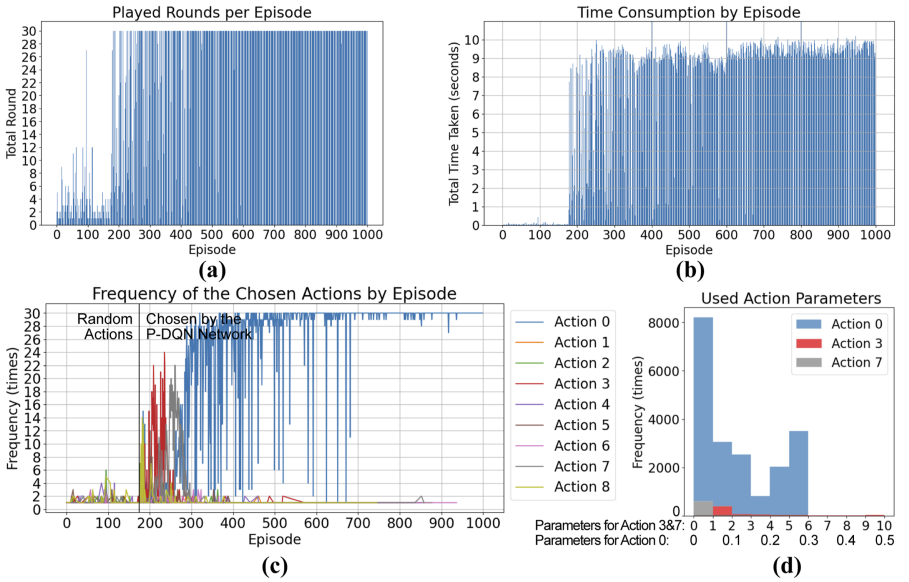
**Note(s):** The parameters have different meanings in the binary version of the game
**Source(s):** Authors' own work

## 2. Detailed Result Data



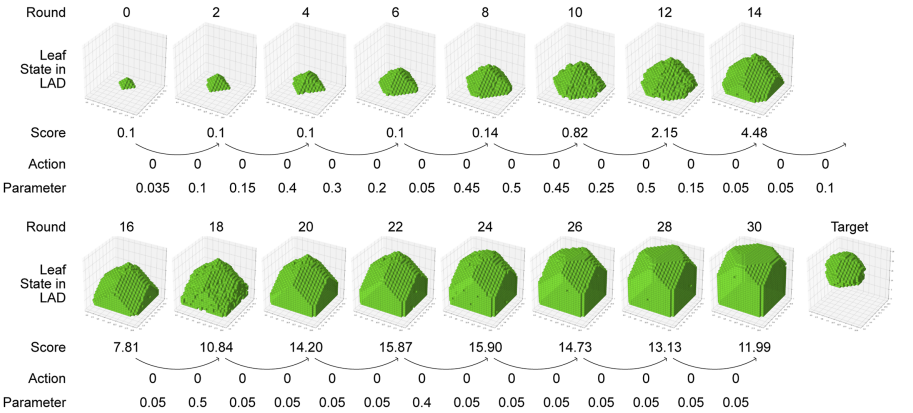

**Source(s):** Authors' own work

**Figure A1.** Statistics from the log file regarding the game round, consumed time, and actions chosen in the experiments




**Source(s):** Authors' own work

**Figure A2.** Statistics from the log file regarding the simplified tree growth game using binary voxels. Compacted information is total rounds, consumed time, and actions chosen for each episode




**Source(s):** Authors' own work

**Figure A3.** Rendered records of the tree growth in the binary tree growth game at Episode 800, where the pruning strategy was stabilized. The starting state at round 0 and the target state were always the same in the training

**Corresponding author**
Qiguan Shu can be contacted at: qiguan.shu@tum.de