GrowSpace: Learning How to Shape Plants

Yasmeen Hitti [1 2] , Ionelia Buzatu [3], Manuel Del Verme [1 2], Mark Lefsrud [1], Florian Golemo [2 4], Audrey Durand [2 5]

[1] McGill University, [2] Mila Quebec AI Institute, [3] Johannes Kepler Universität Linz, [4] Element AI/ ServiceNow, [5] Université Laval

Abstract

Plants adapt over time to their surrounding conditions. We argue that plant responses to an environmental stimulus are a good example of a real-world problem that can be approached within a reinforcement learning (RL) framework. With the objective of controlling a plant by moving the light source, we propose GrowSpace, as a new RL benchmark. The back-end of the simulator is implemented using the Space Colonisation Algorithm. Compared to video game RL environments, this simulator addresses a real-world problem and serves as a test bed to visualize plant growth and movement in a faster way than physical experiments. GrowSpace is composed of a suite of challenges that tackle several problems such as control, multi-stage learning, fairness and multi-objective learning. We provide agent baselines alongside case studies to demonstrate the difficulty of the proposed benchmark.

1 Introduction

Currently there are a limited number of benchmarks that represent real-world systems since they are hard to design and learning from the physical world is difficult [14, 5]. Their complexities stem from high operating costs, their slow movements, and their limited amount of data [4]. Simulators have provided a proxy to real-world systems and have demonstrated success in optimization of control tasks in robotics [18].

We direct our interest on plants, which similarly to robots, need to interact with their environment. Plants are complex and sense their surroundings through actuation and sensing systems [7]. As biological systems, they actuate their movement as a response to an external stimulus such as light [3]. Their spatial reorientation and growth towards light is a tropic response because their movement is influenced by the direction of the light source [13]. Recently, the idea of controlling plant growth through light manipulation has been investigated for the development of bio-hybrid systems such as living structures [21]. The control of a biological agent, presents a set of interesting problems which translate well to the RL community, such as: continuous control [16], multi-stagelearning [24], multi-objective learning [20], and fairness in a multiple plant setting [11].

In this work, we introduce GrowSpace, a new RL environment that enables the control of procedurally generated plant structures. This benchmark is based on real plant responses to light and leverages this response to address a set of diverse challenges that are beyond the scope of bio-engineering. We bring attention to a set of four different challenges that range from classic control to fairness. GrowSpace is an environment that spans across different fields such as plant science, agriculture, RL, and robotics.

The primary contributions of this paper include: (i) An OpenAI Gym-compatible environment [2] for RL, plant science, and robotics research, (ii) the release of 4 different challenges that encompass control, multi-stage learning, fairness, and multi-objective learning, (iii) three baseline agents on the control setting with Proximal Policy Optimization (PPO) [19], Advantage Actor Critic (A2C) [9], and Rainbow DQN [10] with a CNN state encoder, (iv) case studies of the behavior and weaknesses of the agents on all challenges. We do **not** claim that the environment allows for easy transfer of policies to real plants but we argue that this constitutes an important step towards more realistic RL environments, and supports developing agents for noisy biological settings.

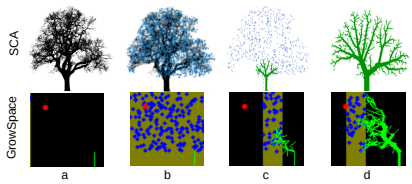

Figure 1: **High-level Overview** of the approach taken for designing the GrowSpace Environment. (a) Determine target. For SCA it is the crown shape of a tree and GrowSpace it is the red circle. (b) Define a point cloud to reach target, SCA scattering surrounds tree crown and GrowSpace is stochastic. In (c) and (d) we show two different growth iterations for SCA and GrowSpace.

2 GrowSpace Learning Environment

The GrowSpace simulator is inspired by a real-world problem of optimizing plant physiology. In the real-world, plant

growth is dictated by several variables, an important one is light availability. GrowSpace incorporates a plant's behavioral response to light and provides control over the branching by means of a mobile light (either light beam or small spotlight). The objective is to guide the growing plant to a desired target or shape depending on the challenge. Figure 1 provides an overview of our approach for designing GrowSpace. The Space Colonization Algorithm (SCA) [17], a branching algorithm, is chosen to mimic a plant's relationship to light. Finally, the branching algorithm is formulated as a RL problem where an agent's objective is to shape a plant towards a target (red) or a desired configuration through means of a mobile light. Further details about SCA can be found in the Appendix A

2.1 Reinforcement Learning Framework

We formulate GrowSpace as a MDP described by a state space $\mathcal{S}$ that is accessed by the agent as a pixel observation, an action space $\mathcal{A}$ that can be discrete or continuous, a transition function $\mathcal{P}$ and a reward function R. On each time step t of a learning episode, the agent observes the state $s_t \in \mathcal{S}$, takes an action $a_t \in \mathcal{A}$, moves to a new state $s_{t+1} \sim \mathcal{P}(s_t, a_t)$, and receives a reward $r_{t+1} \sim R(s_t, a_t, s_{t+1})$. The probability of a plant segments tips to branch in a specific direction given action a_t in state s_t is incorporated into the transition probability $P(s_{t+1}|s_t, a_t)$. In this environment, much like in the real world, the light directly influences the direction of growth of a plant. The agent's objective is to shape a plant towards a target or a desired configuration through means of a mobile and adjustable light source.

States and Observations: For every step taken in the environment, the agent observes the *observation* of its current *state* prior to selecting an action. Once an action is selected by the agent, the new state becomes the observation for the next time step. States and observations are an image representation of the environment which display the plant structure, the light source and the target. The observations are available to the agent as an RGB image that contains the plant, the target and the light beam at time step t. The dimensions are of of $84 \times 84 \times 3$, except for the plant shaping challenge where the dimensions are of $28 \times 28 \times 3$.

Actions: GrowSpace provides a discrete action space and a continuous action space. In the discrete action space the agent can execute five discrete actions. The agent can move the light beam to the right, the left or stay put. The agent can equally increase or decrease the available light beam to the plant. The movement of the light beam is set at a default of 5 pixels in any given direction and can be customized by the user. The continuous action space has two actions, the light velocity, speed at which the light is displaced, and the width of the light beam. This could be a more realistic and more complex set-up, and it will help to transfer the problem from simulation to real world. The actions chosen will influence the available scattering to the plant and will impact the direction of growth of the plant. For example, if the beam of light is not close enough the plant will not be able to branch out because the attraction points and will be dormant. If the light reveals several points, branching will be occur in multiple places in the illuminated area.

In the multiple-objective task, the action set changes due to the circular light beam. Similarly, the agent can increase or decrease the light beam radius, it can move left and right and, can move up and down. The default radius of the beam is 10% of the width of the environment.

Rewards: The reward will be dense and will be received at each time step. Rewards will depend on the challenges in which the agent is trying to solve. Rewards are task specific and explained below in Section 3.

Episode and Reset: The episode length is fixed and is set to 50 steps. At the beginning of each episode, the scattering of photons, and the initial plant stem, as in Section **??**, and the target(s) are procedurally generated in order to ensure the agent will not have visited the exact state previously in other episodes.

3 Tasks

We propose an initial set of tasks that can be tackled in GrowSpace, all of which with several levels of difficulty. The combination of tasks released encompass some known challenges to the RL community, such as control, multi-stage learning, fairness, and multi-objective learning.

Grow Plant to Goal: The task consists in growing the plant with the light beam towards a target positioned at random in the upper 25% of the environment. Every episode begins with the light beam positioned above the original plant stem. The agent must displace the light beam to control and direct the growth of the plant towards the target. After each action, the agent is rewarded based on the smallest distance between any of the branch tips and the target. Let $d_{b,g}$ denote the Euclidean distance between a branch tip b and a target goal g:

$$d_{b,g} = \sqrt{(x_b - x_g)^2 + (y_b - y_g)^2}. \tag{1}$$

The reward obtained at time step t is inversely proportional to this distance of the branch tip closest to the goal among the current branch tips $\mathcal{B}_t$:

$$R_t = \max_{b \in \mathcal{B}_t} \frac{1}{d_{b,g}}. \tag{2}$$

Rewards are therefore in the range $]0, 1[$. This typical control problem [16] is considered the simplest of the tasks since the light movements directly impact the plant from the beginning of the episode. The difficulty of this task is proportional to the distance between the target and the original plant stem tip; as the distance increases, the harder the task becomes.

Find Plant: The task consists in finding the original plant stem with the light source, either the beam or circular light. An episode starts with the light source and the original plant stem positioned at different random locations in the environment. This becomes a multi-stage learning problem [15] where the agent has to first locate the original plant stem by displacing the light source in order to increase the reward signal. The reward is computed using Equation 2. The difficulty of this task is proportional to the distance between the target and the original plant stem tip (as in the Grow Plant task), and to the distance between the original plant stem and the initial light source position. Displacing the light source

multiple times before finding the plant reduces an agent's ability to attain the highest amount of rewards.

Grow Multiple Plants: The task consists in finding two or more plant stems with the light beam and growing them to similar maturities throughout the episode. In this task, the agent must grow $n > 1$ plants towards a target. The target is placed at random in the upper 25% of the environment, the light beam and initial plant stems are initialized randomly within the environment. As in the Find Plant task, the agent must displace the light beam to find all the existing plants in order to initiate a reward signal. Based off our chosen fairness constraint mentioned in section **??** where the objective is to provide equal opportunity and achieve parity amongst plants, the reward signal is crafted as the minimum distance reward (Eq. 2) over all plants:

$$R_t = \min_{1 \leq i \leq n} R_t^{(i)}, \tag{3}$$

where $R_t^{(i)}$ is the grow plant reward (Eq. 2) associated with plant $1 \leq i \leq n$. As seen in Section **??**, different fairness constraints can be adopted in a MDP setting and could be integrated within GrowSpace. We set our first fairness task with a fairness constraint that is similar to [22], which suggests that the agent should provide equal opportunity for each plant to grow towards the target at every step of the episode. The difficulty of this task is in sharing the amount of available photons adequately between plants when they start growing closely to each other. As different plants start approaching each other the photons may run out in the desired direction of the target and the plants may never reach the target (see Appendix E).

Grow Plant to Shape: This task consists in growing plants into specific shapes by using a circular light source that can navigate to precise locations in the environment. As default shapes for benchmarking purposes we consider the MNIST dataset [12], which is widely used in machine learning. MNIST contains 28×28 pixel binary images of handwritten digits (0-9). Given an MNIST image, the goal is to grow a plant such that its shape matches the drawn digit as best as possible. For this task, the environment is reshaped to a width and height of 28×28 pixels (i.e. the size of a MNIST image). The agent has to grow the plant into multiple directions to best cover the outline of the MNIST digit without growing out of bounds. This is a multi-objective task, since the agent has to cover multiple areas in any order, while also keeping the overall goal of limiting the amount of branching in mind.

The reward for this task is crafted using the Jaccard Index [6] similarity score. Let $\mathcal{A}_t$ and $\mathcal{G}$ respectively denote the set of pixels that the plant occupies at time steps t and the set of pixels that belong to the target shape. The reward at time step t is given by the similarity score:

$$R_t = \frac{\mathcal{A}_t \cap \mathcal{G}}{\mathcal{A}_t \cup \mathcal{G}}. \tag{4}$$

4 Experiments and Results

In this section we demonstrate how GrowSpace is challenging by comparing RL baseline algorithms and through a set of case studies.

4.1 Baselines:

We present the different gradient-based policy methods we have evaluated on the easiest challenge in GrowSpace, the control challenge (Grow Plant to Goal task). We compare Proximal Policy Optimization (PPO) [19], Advantage Actor Critic (A2C) [9], and Rainbow DQN [10]. For each of these agents, a state is represented by a tensor of $(3, w, h)$ where w and h are the width and height of the observed image in the task. These representations are fed through three convolutional layers, a fully connected layer and a final layer using the ReLU activation function. The output of the policy network is a probability of each action belonging to the action space. The mean episodic reward over 1 million steps is used to display the learning behaviour of the different RL agents.

In Figure 2 we can observe that PPO displays a greater learning behaviour compared to A2C and Rainbow DQN. PPO was the most promising strategy and was selected as our main baseline to conduct our case studies. We therefore conducted a hyperparameter search for PPO across all challenges and with three different seeds. The details of the final chosen PPO parameters can be found in Appendix B. We equally provide computing times in Appendix D to demonstrate that GrowSpace is a fast running benchmark, similar to Atari.

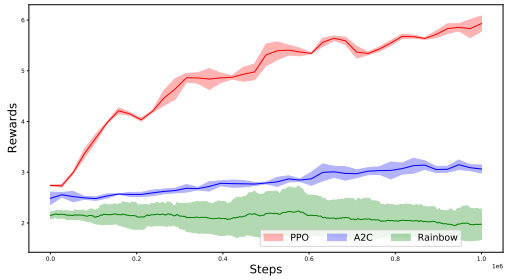

Figure 2: Baseline comparisons for the control challenge

4.2 Case Studies

We present a set of case studies with our main baseline PPO [19] to showcase the spectrum of behaviors of the agent and where challenges are shown to be difficult. For each challenge, we provide an easy and a hard case study, to be described below. Figure 3 displays the initialization of an episode for all challenges in easy and hard settings as well as the the cumulative rewards (averaged and one standard deviation). To better understand the performance of PPO, a random agent and an oracle agent have been implemented for each challenge. No training was performed for the random and oracle agents. The random agent selects actions uniformly at random from the action space. More information about the oracle solutions can be found in Appendix C. The mean episodic reward is our performance metric to evaluate if learning is successful. We include other metrics such as the selection of actions and the overall number of branches produced throughout an episode, these can be found in Appendix A.Results are always averaged over three runs (different random seeds).

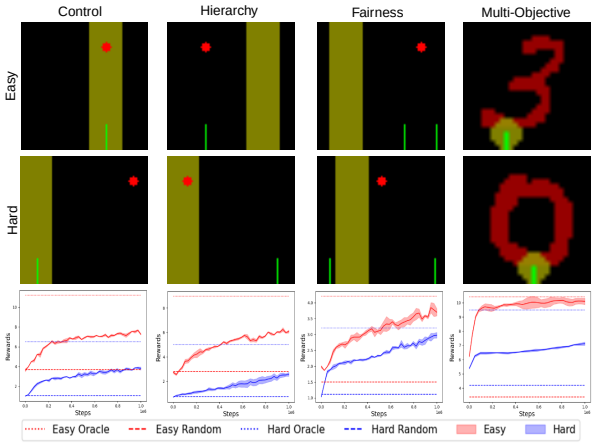

Figure 3: Episode initialization for all case studies

Control: The easy setting is when the target is above the original plant stem and the hard setting is when the stem and target are at opposite extremities of the environment. Figure 3 shows that the easy control reward curve from PPO is closer to the oracle solution and that learning can be improved. For both settings we observe that the PPO reward curve is midpoint between the oracle and random action selection, suggesting that PPO's behaviour can be optimized further. The video renderings show the agent displacing the light away from the plant too quickly, loosing steps with stagnating rewards instead of growing new closer branches and results in guiding the plant to target. The episodic action selection as seen in Figure 6 in Appendix C demonstrates that the agent does not favor decreasing the light beam resulting in a plant with multiple branches competing for the same photons in the direction of the target and thus resulting in slower growth and lower rewards. The action distribution in the easy setting is relatively similar amongst actions, however in the hard setting it is noticeable that the right and left actions are used more.

Multi-stage Learning: Similar to control case study settings with the exception that the light is placed at random at initialization. Figure 3 shows that the hard setting has a lower reward due to the distance between the initial plant stem and the target. With the initial task of finding the plant first, the low reward in the hard setting can be explained by the agent receiving the same reward while trying to find the plant and, the greater distance between the target and the initial stem. The action of increasing the light is more utilized within the harder setting to find the initial plant stem (see Figure 7 in Appendix C). With the video renderings, we the light width does not change a lot once the initial stem is found and the agent learns to drag the light towards the target. The video renderings show that the plant gets bushy and the smaller light width is not utilized efficiently to reduce competition amongst branches for available photons (see Appendix E).

Fairness: The easy setting starts with the episode initialization with both plants at a distance equivalent to the light width and the target is placed in the middle. For the hard setting, the initialization of an episode starts with the plants at the opposite extremities of the environment and the target is placed in the middle of the environment. This case study is particular because the plants are very close and competing for available photons in order to reach the target. As a fairness challenge, the objective is to produce plants of similar size. Figure 3 shows that the easy fairness reward curve from PPO produces the highest amount of rewards. Both PPO reward curves are between closer to the oracle bound than the random agents for both cases. We investigate if the agent's behaviour is fair by tracking the median amount of branches per plant, where the numbers are similar (see Figure 8 in Appendix C. The easy case produces less branches, this can be explained by the small pool of photons that are available to both plants branching and thus limiting additional branching in the right direction. In the middle case, the branching is higher and can be explained by the greater amount of photons available to both plants while reaching the target as they do not need to compete for the majority of the episode.

Multi-objective Learning: The easy settings is that of digit 3 and the hard setting is that of digit 0. The episode initialization for respective case studies starts with the tip of the initial stem touching the MNIST digit and the spotlight placed at random. The easy and hard digits were determined by running PPO individually on all MNIST digits separately. The digits were compared by their median reward values from PPO as seen in Figure 4 and were ranked from easiest to hardest respectively: 3, 6, 2, 1, 4, 5, 7, 8, 9, 0 . Figure 3 shows that the easy multi-objective reward curve from PPO produces the highest amount of rewards. The video renderings show the agent never fully succeeding at filling in the easy digit and the hard digit suggesting that PPO's behaviour and the can be optimized further (see Appendix E). This highlights the difficulty of this challenge and that there is room for improvement for the oracle strategy.

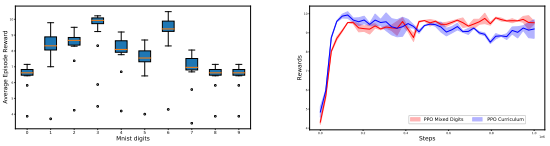

Figure 4: Comparison of digits to design the curriculum for training

An additional case study was explored with the ranked MNIST digits. A curriculum was constructed starting with the easiest digits and consists of 2000 episodes with the two first easiest digits and for every increment of 2000 episodes a new new digit is added. The last 6000 episodes of training have all the MNIST digits. In Figure 4 the learning seems at a higher rate in the first episodes of training for the curriculum approach however, the reward curve decreases as digits are added. The random selection of digits seems to be a better fit over time. We can see that the agent fluctuates the light width in the video renderings rather than visit the full trajectories of the MNIST digits.

References

[1] Biewald, L. 2020. Experiment Tracking with Weights and Biases. Software available from wandb.com.

[2] Brockman, G.; Cheung, V.; Pettersson, L.; Schneider, J.; Schulman, J.; Tang, J.; and Zaremba, W. 2016. Openai gym. *arXiv preprint arXiv:1606.01540*.

[3] Burgert, I.; and Fratzl, P. 2009. Actuation systems in plants as prototypes for bioinspired devices. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 367(1893): 1541–1557.

[4] Dulac-Arnold, G.; Levine, N.; Mankowitz, D. J.; Li, J.; Paduraru, C.; Gowal, S.; and Hester, T. 2020. An empirical investigation of the challenges of real-world reinforcement learning. *arXiv preprint arXiv:2003.11881*.

[5] Dulac-Arnold, G.; Mankowitz, D.; and Hester, T. 2019. Challenges of real-world reinforcement learning. *arXiv preprint arXiv:1904.12901*.

[6] Fletcher, S.; Islam, M. Z.; et al. 2018. Comparing sets of patterns with the Jaccard index. *Australasian Journal of Information Systems*, 22.

[7] Fratzl, P.; Elbaum, R.; and Burgert, I. 2008. Cellulose fibrils direct plant organ movements. *Faraday discussions*, 139: 275–282.

[8] Goyal, A.; Szarzynska, B.; and Fankhauser, C. 2013. Phototropism: at the crossroads of light-signaling pathways. *Trends in plant science*, 18(7): 393–401.

[9] Grondman, I.; Busoniu, L.; Lopes, G. A.; and Babuska, R. 2012. A survey of actor-critic reinforcement learning: Standard and natural policy gradients. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 42(6): 1291–1307.

[10] Hessel, M.; Modayil, J.; Van Hasselt, H.; Schaul, T.; Ostrovski, G.; Dabney, W.; Horgan, D.; Piot, B.; Azar, M.; and Silver, D. 2018. Rainbow: Combining improvements in deep reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32.

[11] Jabbari, S.; Joseph, M.; Kearns, M.; Morgenstern, J.; and Roth, A. 2017. Fairness in reinforcement learning. In *International Conference on Machine Learning*, 1617–1626. PMLR.

[12] LeCun, Y. 1998. The MNIST database of handwritten digits. *http://yann. lecun. com/exdb/mnist/*.

[13] Liscum, E.; Askinosie, S. K.; Leuchtman, D. L.; Morrow, J.; Willenburg, K. T.; and Coats, D. R. 2014. Phototropism: growing towards an understanding of plant movement. *The Plant Cell*, 26(1): 38–55.

[14] Mahmood, A. R.; Korenkevych, D.; Vasan, G.; Ma, W.; and Bergstra, J. 2018. Benchmarking reinforcement learning algorithms on real-world robots. In *Conference on robot learning*, 561–591. PMLR.

[15] Ni, Z.; Paul, S.; Zhong, X.; and Wei, Q. 2017. A reinforcement learning approach for sequential decision-making process of attacks in smart grid. In *2017 IEEE Symposium Series on Computational Intelligence (SSCI)*, 1–8. IEEE.

[16] Recht, B. 2019. A tour of reinforcement learning: The view from continuous control. *Annual Review of Control, Robotics, and Autonomous Systems*, 2: 253–279.

[17] Runions, A.; Lane, B.; and Prusinkiewicz, P. 2007. Modeling Trees with a Space Colonization Algorithm. *NPH*, 7: 63–70.

[18] Rusu, A. A.; Večerík, M.; Rothörl, T.; Heess, N.; Pascanu, R.; and Hadsell, R. 2017. Sim-to-real robot learning from pixels with progressive nets. In *Conference on Robot Learning*, 262–270. PMLR.

[19] Schulman, J.; Wolski, F.; Dhariwal, P.; Radford, A.; and Klimov, O. 2017. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*.

[20] Van Moffaert, K.; and Nowé, A. 2014. Multi-objective reinforcement learning using sets of pareto dominating policies. *The Journal of Machine Learning Research*, 15(1): 3483–3512.

[21] Wahby, M.; Heinrich, M. K.; Hofstadler, D. N.; Neufeld, E.; Kuksin, I.; Zahadat, P.; Schmickl, T.; Ayres, P.; and Hamann, H. 2018. Autonomously shaping natural climbing plants: a bio-hybrid approach. *Royal Society open science*, 5(10): 180296.

[22] Wen, M.; Bastani, O.; and Topcu, U. 2019. Fairness with dynamics. *arXiv preprint arXiv:1901.08568*.

[23] Yandun, F.; Silwal, A.; and Kantor, G. 2020. Visual 3D Reconstruction and Dynamic Simulation of Fruit Trees for Robotic Manipulation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 54–55.

[24] Yang, Z.; Merrick, K.; Jin, L.; and Abbass, H. A. 2018. Hierarchical deep reinforcement learning for continuous action control. *IEEE transactions on neural networks and learning systems*, 29(11): 5174–5184.

A GrowSpace Branching Algorithm

The SCA [17] model is chosen for GrowSpace due to its success in generating virtual trees with features that are comparable to real trees[23]. Through the attachment of plant segments to a plant structure, this algorithm facilitates the iterative growth of a virtual plant.

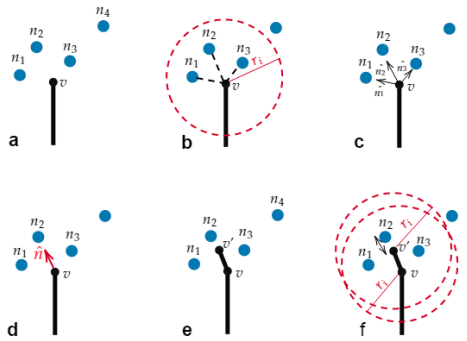

Figure 5: Steps for branching in the Space Colonization Algorithm, where (a) all photons are filtered (b) trough a radius of attraction (c) and their normalized vectors from the plant tip to the photons (d) are summed and normalized to find the direction of growth (e) for the new plant segment to be attached (f) process is repeated for all existing plant tips

In Figure 5 (inspired by [17]) illustrates the algorithm. The algorithm begins with a set of photon particles N and an initial plant segment with tip v (a). The plant segment tip eventually become a set as the plant grows, where $v \in V$. In order for a plant tip to grow, photons $n \in N$ must be located within a predefined radius of influence r_i, as seen in (b) where $n1$, $n2$ and $n3$ attract segment tip v. When a photon is too close to a plant segment, the photon is removed and is not considered. The normalized vectors from tip v towards photons $n \in N$ are computed (c). Once summed, the normalized vector $\hat{n}$ is found for v (d). The vectors representing the direction of growth are:

$$\overrightarrow{n} = \sum_{n}^{N} \frac{n - v}{\|n - v\|}. \tag{5}$$

The final normalized vector for a plant segment tip is:

$$\hat{n} = \frac{\overrightarrow{n}}{\|\overrightarrow{n}\|}. \tag{6}$$

Vector $\hat{n}$ is the direction of growth of plant segment tip v and is towards photon n_2. The plant grows a new segment v' (e). The procedure is then repeated on both of the plant segment tips. In (f) we can observe that n_2 is too close to v' and will not be considered for branching. v In GrowSpace the attraction points are thought of as available photon particles and are scattered at random to facilitate stochastic branching. The amount of observable photons are limited and available to the plant with a light source. The light source illuminates photons within a certain range (that are not visible to the agent) and consequently restricts the direction of growth. In Figure 1 we enable the scattering of photons to be visible for better understanding of GrowSpace. We introduce the concept of light direction to encourage unidirectional growth towards the light source. To grow towards the light source, shading needs to take place as to not allow the light beam to illuminate the photons that are below existing parts of the plant foliage. This integration is based on phototropism, a response process, that enables plants to adjust their growth towards the direction of the light [8].

A.1 Parameters

In Table 1 we describe how parameters of GrowSpace can influence the overall output shape of a growing plant. Default values are provided but can be changed by the user.

GrowSpace Parameters	
Light Density	Increases stochastic branching and chance of branching as in increases. If too small a plant may not grow due to a photon never being in the radius of attraction
First Branch Height	This will reduce the amount of branching if increased and facilitates reaching photons that are closer to the target. If decreased, the plant can become bushy quickly and may never attain target when light width is not controlled properly.
Max Branching	Upper limit of possible amount of new branches per step, the greater it is the bushier the plant will get, the lower it is, the chance of branching reduces
Branch Length	Length at which every branch can grow. If value is high this results in faster growth. If value is low the plant will grow more slowly and will need more steps to attain the same target.
Light Displacement	Increment at which the beam or focal light can move in any direction. If too big it may skip photons along a desire trajectory, if too small, this may lead to not completing task.

Table 1: Parameters that can be changed by the users in GrowSpace

B Hyperparameter Search

A first hyperparameter search was performed across GrowSpace parameters that can be customized by the user. The selection of default values for GrowSpace were chosen by qualitatively assessing the renderings of plants and selecting the combination of values that created realistic virtual plants. For each GrowSpace parameter we set a range of values. Values ranging from $[0, 1]$ are multiplied by the default resolution of the environment of 84 pixels for the control, multi-stage and fairness challenges.

GrowSpace Parameter	Min	Max	Final
Light Density	100	400	200
Initial Light Width	0.01	1	0.25
First Branch Height	0.05	0.5	0.2
Max Branching	1	20	8
Branch Length	0.033	0.3	0.1
Branch Thickness	0.015	0.05	0.015

Table 2: GrowSpace default parameters after performing a sweep

The hyperparameter sweep for PPO was performed across 3 seeds and values were tested on all 4 challenges. The average episode reward was the metric for determining which combination of hyperparameters allowed for better learning. It is interesting to note that the final hyperparameters are similar to the ALE benchmark for PPO.

PPO Hyperparameter	Min	Max	Final
Learning rate	0.00001	0.1	0.00025
Epsilon	0.01	1	0.25
Gamma	0.05	0.99	0.99
GAE lambda	0.3	0.99	0.95
Entropy coefficient	0.01	0.5	0.01
Max grad norm	0.1	0.9	0.5
Number of steps	1000	5000	2500
PPO epoch	1	20	4
Mini batch	10	100	32
Optimizer	adam	sgd	adam
Momentum	0.95	0.99	0.95
Clip parameter	0.05	0.5	0.1

Table 3: Range of PPO hyperparameters used while tuning and final values

C Additional Details on Experiments

Oracles: For the control and multi-stage challenges, the target location is known before hand and the light is moved accordingly until the plant has reached the target. For the fairness challenge, the oracle first evaluates if the location of the target is in between, on the right or the left side of the two plants. Once known, the oracle displaces the light to the furthest plant and alternates the lighting to achieve similar growth in both plants towards the target. For the multi-objective challenge, the oracle knows the desired digit to form and displaces the light to fill the digit from bottom to top all while keeping the light within the MNIST digit pixels.

Action selection: The following are action distributions for the control and multi-stage challenges for the case studies discussed in Section 4.2.

Branching: In figure 8 we quantitatively validate if the agent is being fair in growing both plants we look into the

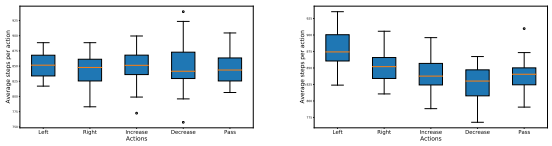

Figure 6: Action selection during training for control case studies

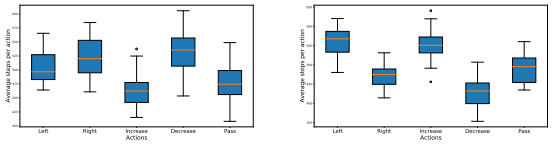

Figure 7: Action selection during training for multi-stage case studies

average number of branches for each plant over the episode.

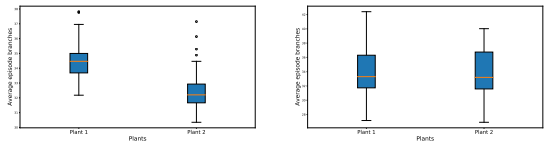

Figure 8: Easy and hard cases episode branching for fairness challenge

D Compute Power

The total amount of compute for this project was of 322 days. This is in part due to game development and the two hyperparameter sweeps done on GrowSpace and PPO. Training was done with 1 GPU and 8 CPU and an average run depending on the challenges in GrowSpace can take from 1 hour and 30 minutes to 3 hours. More details can be found in the reports shared below in Appendix E. These computing times for one run is comparable to Atari.

E Viewing Agent Videos and Experiments

Agent videos and experiments have been tracked with weights and biases [1]. Individual dashboards were made for easy access and to better compare case studies within challenges.

- Control challenge: https://wandb.ai/growspace/control/reports/Control-Challenge--Vmlldzo3NTk1NDk
- Multi-stage challenge: https://wandb.ai/growspace/hierarchy/reports/Hierarchy-Challenge--Vmlldzo3NTk1MzI
- Fairness challenge: https://wandb.ai/growspace/fairness/reports/Project-Dashboard--Vmlldzo3NTk1NjI
- Multi-objective challenge: https://wandb.ai/growspace/mnistexperiments/reports/Project-Dashboard--Vmlldzo3NTk1NTk

The Machine Learning Reproducibility Checklist (v2.0, Apr.7 2020)

For all **models** and **algorithms** presented, check if you include:

- ☑ A clear description of the mathematical setting, algorithm, and/or model.
- ☑ A clear explanation of any assumptions.
- ☑ An analysis of the complexity (time, space, sample size) of any algorithm.

For any **theoretical claim**, check if you include:

- ❑ A clear statement of the claim.
- ❑ A complete proof of the claim.

For all **datasets** used, check if you include:

- ❑ The relevant statistics, such as number of examples.
- ❑ The details of train / validation / test splits.
- ❑ An explanation of any data that were excluded, and all pre-processing step.
- ☑ A link to a downloadable version of the dataset or simulation environment.
- ❑ For new data collected, a complete description of the data collection process, such as instructions to annotators and methods for quality control.

For all shared **code** related to this work, check if you include:

- ☑ Specification of dependencies.
- ☑ Training code.
- ☑ Evaluation code.
- ☑ (Pre-)trained model(s).
- ☑ README file includes table of results accompanied by precise command to run to produce those results.

For all reported **experimental results**, check if you include:

- ☑ The range of hyper-parameters considered, method to select the best hyper-parameter configuration, and specification of all hyper-parameters used to generate results.
- ☑ The exact number of training and evaluation runs.
- ☑ A clear definition of the specific measure or statistics used to report results.
- ☑ A description of results with central tendency (e.g. mean) & variation (e.g. error bars).
- ☑ The average runtime for each result, or estimated energy cost.
- ☑ A description of the computing infrastructure used.

Reproduced from: *www.cs.mcgill.ca/~jpineau/ReproducibilityChecklist-v2.0.pdf*