# A Additional Details on GrowSpace

In Table 2 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 2: 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, hierarchical 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 3: 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.

# C Baseline Comparison

In Fig. 5 we can see the performance of 3 algorithms (PPO, A2C, Rainbow) on the control challenge with randomized initialization. We can clearly see that both A2C and Rainbow were struggling to solve the task, whereas PPO showed steep learning behavior after a few thousand steps in the environment.

| 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 |
| Generalized Advantage Estimation 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 4: Range of PPO hyperparameters used while tuning and final values

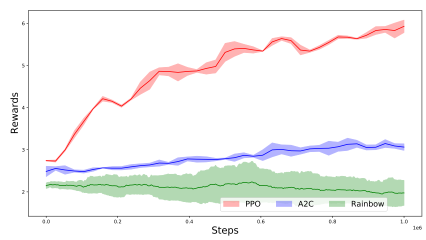
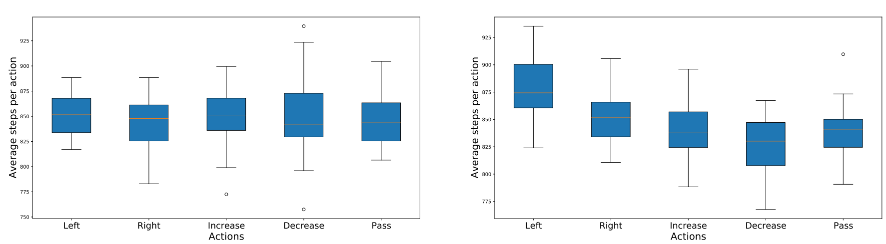



Figure 5: Baseline comparisons for the control challenge

# D Additional Details on Experiments

**Oracles:** For the control and hierarchical 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 hierarchical challenges for the case studies discussed in Section 6.1.

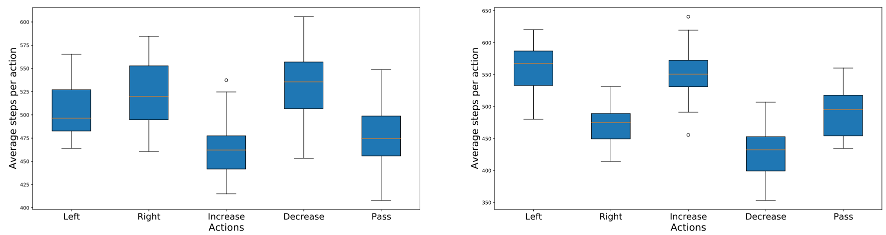



(a) Action distribution easy

(b) Action distribution hard

Figure 6: Action selection during training for control case studies

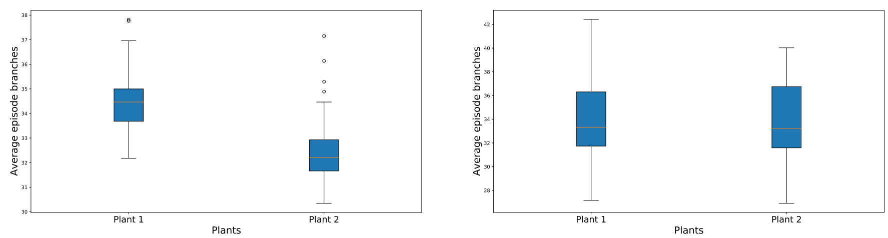
**Branching:** In figure 8(b) we quantitatively validate if the agent is being fair in growing both plants we look into the average number of branches for each plant over the episode.




(a) Action distribution easy

(b) Action distribution hard

Figure 7: Action selection during training for hierarchical case studies




(a) Fairness close challenge

(b) Fairness above challenge

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

# E 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 1GPU 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 F.

# F Viewing Agent Videos and Experiments

Agent videos and experiments have been tracked with weights and biases [2]. 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`
- Hierarchy 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:

- [x] A clear description of the mathematical setting, algorithm, and/or model.
- [x] A clear explanation of any assumptions.
- [x] 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.
- [x] 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:

- [x] Specification of dependencies.
- [x] Training code.
- [x] Evaluation code.
- [x] (Pre-)trained model(s).
- [x] 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:

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

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