Agricultural Biosystems Engineering Group

MSc Thesis Agricultural Biosystems Engineering

# Evaluation of DreamerV3 for defective plant search in a simulated crop

A General-Purpose Agent: Saviour of our Simulated Crops

Tim Bonte

Date 05/03/24

WAGENINGEN
UNIVERSITY & RESEARCH

## Summary:

The use of autonomous robots is increasing in agriculture; however, their development is still slow as each task usually needs a specific hard-coded behaviour. Hard coding of all behaviors will be long and might lack flexibility and optimality. Reinforcement learning (RL) seems a promising approach to address these issues as it could learn these behaviours. However, classical RL agents struggle to work with high-dimensional inputs, which is usually solved by manually extracting features which can again be suboptimal, and biased. This study will make use of a general-purpose deep model-based reinforcement learning agent to reduce the dimensionality of the inputs and obliviate the need for hard-coded feature extraction while learning to perform a goal-directed task. Moreover, tasks are varied, they are subject to the environment, the reward, the inputs, etc. A general-purpose algorithm is a reinforcement algorithm that aims to perform any task with fixed hyperparameters, obliviating the need for optimization and human intervention.

This research explores the potential of a general-purpose deep model-based RL algorithm, the DreamerV3, for agri-food robotics. This study focuses on the context of crop defect search using autonomous drones in a simulated environment. Drones offer a cheap and precise solution for crop management with a bird's eye view. The methodology includes training and testing the world model's ability to abstract the world. The detection rate of the actor-critic to search for defects in crops will then be evaluated, in the case of single and multi-patch distribution of the defects using solely the image or increased with a cover map. All scenarios will be compared to the baseline comparison, a traditional row-by-row flight path.

The findings of this study have the potential to contribute to the development of autonomous robots for agriculture, improving efficient crop management but more importantly, evaluating the potential of highly flexible algorithms to perform different tasks.

# 1 Introduction

## 1.1 Current situation

Since the end of the Second World War, the world has witnessed a significant surge in population. This demographic explosion has exerted tremendous pressure on the global farming system to boost production (Bocquet-Appel et al., 2012; Ehrlich, 1995). Labour shortage exacerbated by migration to urban areas and changing societal roles, has made it increasingly challenging for farmers to meet the growing demand for food (Ryan et al., 2023). Nowadays the population is still growing (CBS, 2023) and is expected to follow an ascending trend. Automation in agriculture had a significant impact on production levels since 1950 (Steenwyk et al., 2022), as example, the introduction of tractor tripled the production. This automation not only boosted production but also started the escalating trend toward monoculture fields, enabling the use of larger machinery. Those have also brought their side effects such as soil compaction or CO2 emission when deeply tilling (Mooney Sacha et al., 2023).

The relentless depletion of natural resources, such as arable land, freshwater, and fertilizer, has endangered the sustainability of our farming practices (Ehrlich, 1995). Moreover, intensive farming practices have generated collateral damage. Leaching fertilizer brings eutrophication leading to toxic aquatic (Doster et al., 2013) and foreshore environments (Gladyshev & Gubelit, 2019). Furthermore, the soil erosion caused by deforestation (Moisa et al., 2022), the loss in biodiversity contributing to the intense use of pesticides (Isenring, 2010), and substantial production of greenhouse gas (GHG). In the global effort to reduce GHG, the food system represents, from cradle to gate, the third of the global emissions (Crippa et al., 2021), where 71% arise from agriculture. The shift in regulations is parallel to the shift in the market trends with an increasing demand for environmentally friendly products (Rehder, 2022).

Faced with environmental and societal challenges in agricultural production, the sector urgently needs innovative solutions to ensure both food security and environmental preservation in our fast-changing world. Recent technological advancements offer potential solutions through smart farming, an approach integrating technology, data, and automation for monitoring and optimizing crop production. While these innovations have the potential to revolutionize how farmers feed the world, they also necessitate new skills like data management and hardware maintenance. Despite significant strides in automation for tasks such as path planning (Bai et al., 2023), greenhouse management (Zhang et al., 2021), harvesting (Van Henten et al., 2002), pruning, and disease/pest management, manual labor remains prevalent. This reliance on manual work can be time-consuming and expensive, with the added challenge of human error. Moreover, the demand for skilled human labor persists, and sometimes shortages in European countries (Ryan et al., 2023)

In recent years unnamed aerial vehicles (UAV) such as drones have displayed great potential for farmers. They are nowadays in increasing use in farms (Rejeb et al., 2022) for crop monitoring, pest management (Hafeez et al., 2023), seeding or irrigation. Drones can on one hand be precise in their action, limiting the usage of resources and their impact on the environment. On the other hand, they should decrease the time spent on crop monitoring powered by their flexibility and rapidity. Drones for agriculture have also displayed significant challenges (Sharma & Hema, 2021). These challenges can be technical with the battery limiting the working range, and weather dependencies. They can also be societal with regulations or farmer knowledge and skills such as the ability to control the drone.

Introducing automation through autonomous robots poses significant challenges due to the complexity and multitude of tasks involved. The conventional approach often requires breaking down tasks into sub-components, each requiring specific hard coding and integration. Unfortunately, this can result in agents lacking adaptability and generalization, particularly in unpredictable environments. To overcome these challenges, a promising avenue is the application of Reinforcement Learning (RL). RL offers the potential to learn goal-directed tasks without explicit programming, addressing the limitations associated with hard-coded approaches. This shift towards RL signifies a key advancement in enhancing the flexibility and adaptability of autonomous agents in dynamic environments.

Controlling navigation in a field for crop monitoring has received some attention in recent years, with the development of coverage path planning methods Fevgas et al., 2022). Their goal is to cover an entire area of interest while minimizing overlapping. The simplest pattern is the Boustrophedon (Figure 1), also known as Row by Row (RbR). It consists of a simple back-and-forth movement over the field. Many more hard-coded path-planning methods can be used (Fevgas et al., 2022). Other approaches such as the travelling salesman problem, finding the shortest path for visiting desired locations could find their use however they require prior knowledge of the location to visit.

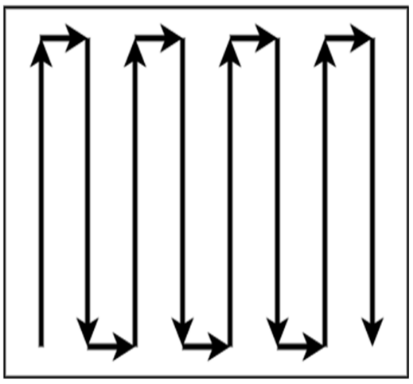


*Figure 1 Row by Row Path (Fevgas et al., 2022)*

Defect detection such as disease, nutrient deficiencies, etc, had a lot of attention in recent years with the development of phenotyping or enviro-typing using machine vision and mainly using convolutional neural networks (Cun et al., 1989). Those tasks are usually optimized for the plant level identifying whether a leaf or a plant is defective or not from an image. However, applications for farmers' crops are more limited and might require special needs such as bird eye view, and specific datasets (etc). A possible current application is realized using satellite data. However, access might be limited due to the orbit, the weather or cost with lower temporal and spatial resolution (Sylvester, 2018) which makes it poor use for the farmer. Effective defect detection in agriculture can significantly impact farms' profitability, multiple types of defects can undermine crop health. Failure to act in detecting and addressing these defects correctly and on time can lead to severe consequences, including substantial losses in crop yield and financial setbacks for farmers.

## 1.2 Desired situation

Smart farming is a developing trend in agriculture with the integration of intelligent robots full of sensors in farms (Said Mohamed et al., 2021). This offers a remarkable opportunity to reduce costs and optimize production with a strategic focus on achieving precise operations in the near term, while also pursuing sustainable agricultural practices and ecological. In terms of crop management, the goal is to have autonomous, continuous, and precise monitoring leading to a precise impact on the crop state in the early stages of defect. This would reduce resource usage and impact the environment while maximizing yield. On another hand, robot hardware should have a minimal impact on the environment, and soil quality through soil compaction or excessive resource usage, as well as be affordable. An intelligent spraying drone seems a good option to fulfil the role of monitoring the crop's state. As it is flying it could investigate the field rapidly and take precise actions. Drones provide a bird's-eye view of crop fields, making it easier to monitor crop growth and identify stressed or defective plants, in early signs of disease or nutrient deficiencies. Drones are also affordable compared to satellite images used currently.

In terms of path planning for crop monitoring by drones, the desired situation is to inspect the crop and find all defective plants in minimal time. This goal could be considered to lie between the coverage path planning and the travelling salesman's problem. The most optimal is the result of a travelling salesman problem, as it tries to find the shortest path for a tour however it requires knowing the position that needs to be visited which is not the case. It is therefore desired that the agent proactively searches its environment to find the defects while doing so in the shortest path possible. This demands active perception, as the agent must dynamically navigate and interact with its surroundings to complete its task and gain information about the localization of defective plants.

Extending this discussion to the broader perspective of robotic autonomous software, the aspiration is to develop flexible and intelligent algorithms capable of adapting seamlessly to various robots, tasks, and environments with minimal human intervention. That includes continuous and discrete actions, visual and low-dimensional inputs, different reward scales & frequencies, and supporting 2D and 3D environments. This is commonly called general-purpose algorithms and has long been a goal of reinforcement learning research. It would limit human input, on the one hand by limiting the optimization of hyperparameters between tasks. On the other hand, it would limit hard coding tasks while building accurate abstraction of the world allowing to make correct decisions through a learned process. Such technology could be a novel method as it could speed up the development of autonomous robots for agricultural purposes.

## 1.3 Problem definition

We have identified significant potential for the development of drones for agriculture, focusing on the primary task of exploring a field to find defective plants in minimal steps. In this study, we aim to address technical challenges arising from hard-coded path generation, humanly engineered feature extraction, and autonomous exploration for defective search in crops.

The currently used hard-coded path planning lacks flexibility and optimality. Although coverage path planning covers the whole field, it assumes the objects to be distributed uniformly and pays equal attention to all parts of the field, while crop defects are emerging from patches. This is where learning what action to take in a certain state could resolve this problem. The suboptimality aspect arises from the repartition of the defect, as it is unlikely that they are uniformly distributed over the field (Ash, 2021; Been et al., 2022). The distribution of defects in fields is dependent on the type of defects such as pests, viruses, or water depletion (Vacante & Bonsignore, 2012). For vector-based defects, it is likely that they arise from an epicenter from which the contamination arises, which would make the distribution more Gaussian (Been et al., 2022). This distribution should reduce the need to search the entire field as it would be needed to locate the epicenter. Their inflexibility comes from the single-shot aspect and independence from inputs of the methods employed, once the path has been defined the agent will execute the commands independently of the surrounding environment. Its absence of adaptation to the current state is a critical problem as it might cause damage to the crop and the drone countered by increasing caution when deployed. Their inflexibility might also affect the optimality of the methods as it will not shorten the path as it is a finite path length. The central challenge lies in the unification of detection and exploration within the context of traditional path planning methods, which often lack awareness of the goal they aim to achieve. To achieve precise and efficient crop protection through active perception, it is imperative to master both defect detection and optimal navigation in crops. Consequently, the key solution involves empowering the agent to explore its surroundings while employing active perception to search for defects in the crop. The core lies in consistently striking a balance between goal-oriented decision-making and exploration, ensuring a seamless integration of these two critical aspects to enhance the overall effectiveness of crop protection measures.

RL can be a solution that can increase the adaptability of path planning by learning the action guided by the reward and from the representation of the environment, the state. However, the agent is constrained by the different components of its environment. As the input size on which the agent takes the decision increases, the agent requires more data to map the action to the state. Essentially RL agents cannot directly work with high-dimensional states. This problem could be resolved by manually engineering symbolic representations of the world. Currently, those are based on geometrical features, like shapes and positions, or semantic features, like classes of objects and their relationships. As they are humanly extracted features, they are susceptible to what the crafter believes to be relevant, which could bias the training. This is where learning to automatically extract features from the inputs could resolve the high-dimensional state issues. This method offers a solid framework to deal with any type of environment and their input which includes 2D & 3D. Other constraints arise from the use of RL agents, such as the tuning of hyperparameters depending on the type of autonomous agent used. Defining the reward of the agent is also crucial as it targets the direction of the goal. Dealing with different scales and frequencies might be important to reach the desired goal. The problems presented above still require heavy R&D processes, slowing the development of autonomous robots in agriculture.

A latent dynamic model-based RL agent such as DreamerV3 could solve most of the different challenges presented. It would be able to learn to extract key features (latent state) from the inputs, creating a model of the environment, on which it could dream what impact its action would have on the environment (model-based) and then learn what action to select in the specific state (RL).

## 1.4 Objective

The key objective of this study is to evaluate the performance of a latent dynamics model-based RL (DreamerV3) to search for defective plants using a drone in a simulated crop. To do so we want to build an adaptative policymaker to increase the flexibility of autonomous robots by learning the control of a robot from a high dimensional state such as images. It therefore involves the agent learning latent representation from input, learning the dynamics of the environment, and learning to perform the desired task efficiently.

## 1.5 Research question

**Main Research Question (RQ):** How effectively can a latent dynamics model-based RL agent (DreamerV3) control a drone to search for crop defects in a simulated environment?
**RQ1**: What is the performance of DreamerV3 in searching defective plants in a crop with different distributions using a drone?

**Sub-RQ 1.1**: What is the error of DreamerV3's world model reconstruction for drone control to search for crop defects?

**Sub-RQ 1.2:** What is the detection rate of defective plants in a simulated crop using DreamerV3 while exploring a crop using a drone?

**RQ2:** What is the performance of DreamerV3 in searching defective plants in a crop with different distributions and increased context using a drone?

**Sub-RQ 2.1:** What is the error of DreamerV3's world model reconstruction for drone control to search for crop defects?

**Sub-RQ 2.2:** What is the detection rate of defective plants in a simulated crop using DreamerV3 while exploring a crop using a drone?

## 1.6 Demarcation

The research will remain theoretical and simulation-based, with no direct application to real-world robotic deployment considered within this work. The environmental parameters established at the beginning of this research will remain constant; there will be no modifications to the experimental environment. As such, the impact of environmental factors on the robot's performance will not be assessed. The evaluation will focus on the performance of a single crop. The impact of the type of defects will not be assessed. This thesis will not engage in comparative analysis with other RL methods. The research will not explore the effects of scaling the neural networks sizes on performance outcomes.

## 1.7 Outline

This thesis focuses on the evaluation of 2 main scenarios compared based on their ability to find defective plants in a simulated crop. Section 2 will make an overview of RL methods to introduce the Dreamer architecture. Section 3 will display the materials and methods used in our research. Section 4 will display the result initially focusing on the world model performance and then the actor critic's ability to perform the desired task. Section 5 will discuss the results. Section 6 will conclude by answering the research questions. Section 7 will make recommendations and suggestions for further research.

# 2 Background

Machine learning, a branch of artificial intelligence, has enhanced the state-of-the-art performance of autonomous agents and the way we approach problem-solving in various fields. At its core, machine learning involves the development of algorithms that enable computers to learn from and make predictions or decisions based on data. This approach contrasts with traditional programming, where explicit instructions are given for every scenario. Machine learning algorithms improve their performance as they are exposed to more data, adapting their responses based on patterns and insights derived from this data. Within this domain, several paradigms have emerged, each with unique methodologies and applications.

In the 20th century, the renowned psychologist B.F. Skinner introduced operant conditioning (OC); this foundational principle of behavioral psychology revolves around the idea that an organism's behavior is shaped by the outcomes it experiences. In other words, the consequences of an action play a pivotal role in determining whether that action is repeated or suppressed. Operant conditioning is a fundamental process in understanding how living organisms adapt and learn from their environment. Reinforcement learning is one of the paradigms of machine learning and builds upon the concept of operant conditioning as it aims to train a software agent to learn what to do in a defined situation so that it maximizes cumulative rewards. Based on Sutton & Barto (2018) almost any goal-direct decision process can be seen as a Markov decision process (MDP) (Figure 2), this concept depicts the core of reinforcement learning.

## 2.1 Markov decision process (MDP):

The MDP framework (Figure 2) structures decision-making (Sutton & Barto, 2018) by breaking it down into three essential communication channels between an agent and its environment, those are the action, the state, and the reward.

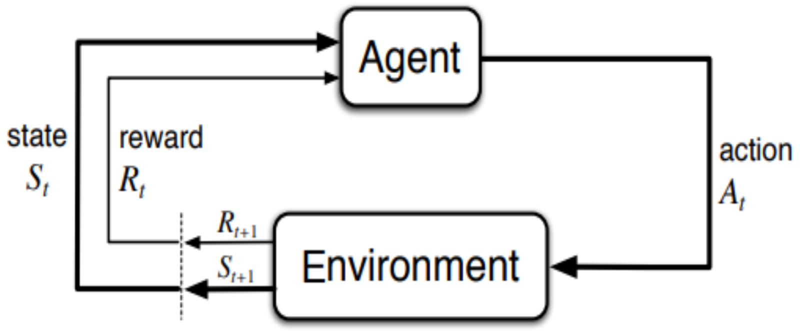


*Figure 2 Markov decision process (Sutton & Barto, 2018) (Agent and environment are the entities and the arrow are their communication channel, state, reward, action)*

An agent learns and decides what action to take, this agent is put in an environment, where whatever surrounds the agent learning is considered the environment. At each step the agent receives a representation of the environment, the state.
The agent will interact with this environment through an action, the policies, taken from the possible actions, the action space. This action will have an impact on the environment, leading to a change in the state of the perceived environment and a response of the

environment to this action, the reward. The goal of this agent is to maximize the total reward received therefore the reward is crucial as it will constrain the goal of the agent, as well as not prioritizing immediate rewards. A state is Markovian when the future is independent of the past given the present. This means that the future is solely dependent on the present state which captures all history. For any Markov decision process, it is therefore possible to compute the state transition function which holds the likelihood of transferring from one state to the other. While this framework might not encompass all decision-learning scenarios, it has demonstrated its flexibility and practical applicability.

Reinforcement Learning (RL) uses the MDP framework to learn a policy that makes decisions. RL is usually categorized into model-free / model-based and on / off policy RL. Model-free RL learns from the raw input and Model-based plan using an abstract representation of the world, the model. On-policy RL collects data while following the control policy and off-policies RL follows a separate policy while learning a control policy. Figure 3 displays the different approaches used in the RL world using the model-free/model-based classification, this classification will be used to depict the RL algorithm in more detail.

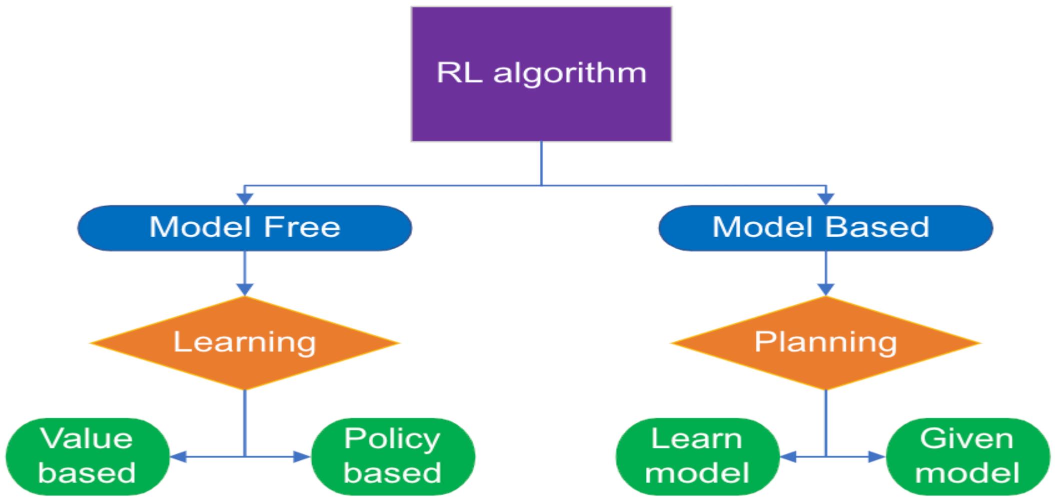


*Figure 3 RL algorithm classification (The first classification in blue, each classification focuses on a task in orange, and the final in-classification is then depicted in green)*

## 2.2 Model Free Reinforcement learning

Model-free strongly relates to the basic idea brought by Skinner where an agent uses trial and error to learn the outcome of its actions. That agent relies on exploration of their environment, acting on the environment to find the most rewarding behavior in the different states. The focus of this agent is on learning (Figure 3) what action to take knowing the current state. However, for traditional RL the trial-and-error mechanism requires that the agent visit the possible state to update the probability of repeating the policy. As the state-space is increasing it becomes expensive to explore all possible states. Their efficiency is therefore limited as it requires tremendous amounts of iterations in a simulated environment exponentially increasing with the size of the state.

Model-free focuses on learning, the question is learning what, from which two approaches arise. The first one and one of the most popular is Value-based such as Q learning (Watkins & Dayan, 1992). Q-learning aims to learn a Q function which stores the estimated expected

cumulative rewards (Q-values) for taking a specific action in a specific state. The key idea is to learn a state transition probability and reward table. The Q-table is therefore finite in a deterministic environment and depends on the state (S) and the actions (A), once all states and actions have been explored, the training has converged. However, for high dimensional input this table can be exceedingly long to fill in as it requires S*A iteration.
The second approach, policy-based (Sutton et al., 1999), focuses on directly optimizing the policy. By iteratively adjusting the parameters of the policy, moving in the direction that will improve the expected returns, and therefore directly maps the state to action (Chadi & Mousannif, 2023). This method has proved to be efficient when dealing with high-dimensional spaces and continuous actions as it allows for more precise adjustments (Zhao et al., 2022).

Both approaches aim to maximize the expected cumulative reward, but they differ fundamentally in their mechanisms: value-based updates value estimates for state-action pairs, while policy-based directly manipulates the probability of acting in a state to achieve better outcomes. Both can be put together to overcome their weakness while leveraging their strength.

## 2.3 Model-based reinforcement learning

In addition to operant conditioning, the behavioral psychologist Edward Tolman introduced that animals develop an internal representation of the world (Tolman, 1948) perceived through their senses without any reward (Ha & Schmidhuber, 2018). This brings us to the second classification which is model-based reinforcement learning. The key addition of these agents is the use of a model of the environment to plan the future state and reward referred to as planning (Figure 3). A model of the environment is a representational function which enables to predict the future state and reward of the agent (Sutton & Barto, 2018) This main addition enables the agents to plan by thinking ahead.

Traditionally, agents relied on symbolic representations, meticulously crafted by developers that represent a model of the world. These representations encompassed geometric features like shapes and Cartesian positions or semantic features such as object classes and relationships. However, these handcrafted representations are constrained by human-defined criteria and prior knowledge of the world. Those models, being heavily dependent on assumptions, suffer from bias and require meticulous selections. The performances of such models are constrained by the quality of the abstraction to capture the key feature.

## 2.4 Deep Reinforcement Learning

Over the past decade, there has been a remarkable shift towards data-driven methods with the deep learning revolution (Hajkowicz et al., 2023), reinforcement learning is not an exception. Deep RL makes use of an RL method combined with neural networks. It exists for both the model-based and the model-free path. For example, in the model-free algorithm using Q learning, instead of building a state action table, it is possible to use a neural network to predict the function of the state transition probability.

For model-based deep RL, the main goal is to automate the creation of the model using feature extraction methods such as convolutional neural networks. These networks play a pivotal role in distilling raw, high-dimensional data, such as images, into a more compact and meaningful representation of the environment. Neural networks such as CNNs excel in autonomously identifying and extracting salient features without explicit human-engineered instructions, thus creating an unbiased and generalized model of the world. Although these representations may lack human-understandable semantics, they provide an abundance of information for the completion of complex tasks. This process of feature extraction reduces the dimensionality of the state-action space, increasing the sample efficiency of the RL agent to learn informed decisions (Hafner et al., 2018).

The concept of coupling a neural network to automated feature extraction to an RL agent is referred to as the World model (Ha & Schmidhuber, 2018). It is now possible to abstract the world automatically from an image in a partially visible space as inputs and draw planning of action from it.

### 2.4.1 Original World Model

In this section, the architecture of the initial world model will be depicted and explained. This agent model is composed of three sub-models whose connection can be seen in (Figure 4). First, the vision model uses a Variational Autoencoder (VAE) to compress each image into a latent space Z. It is the feature extraction technique where CNNs (Cun et al., 1989) are used to convolve information enabling the extraction of key features. The second model is the memory (M) model, its role is to predict the future from the current input and previous recurrence and action. It uses a recurrent neural network (RNN) with a Mixture Density Network (MDN) output layer to predict a probabilistic distribution over the next latent space. These models are usually referred to as latent dynamics models as they predict the latent state forward. Mathematically the model will predict equation 1: The probability of the next latent space:

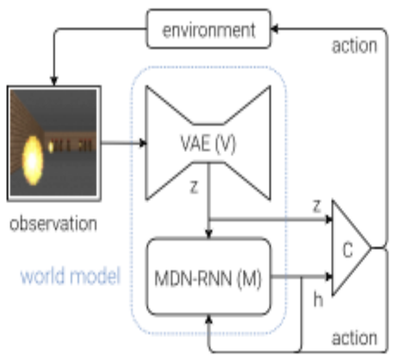


*Figure 4 World model flow diagram (the observation of the environment is encoded by the VAE from which the latent representation is used to propagate by the modern frow ich the controller takes the decision of action) (Ha & Schmidhuber, 2018)*

$$P(z_{t+1} | a_t, z_t, h_t) \qquad [-] \qquad (1)$$

knowing the current state ($z_t$), action $a_t$, and recurrence $h_t$. Finally, the controller model (C) is a single fully connected linear neuron, which learns to maximize the sum of rewards.

### 2.4.2 PlaNet:

Planet (Hafner et al., 2018) build upon the foundation of world models (WM) while leveraging planning, the ability to simulate the consequences of actions. The main difference of a PlaNet is that it does not use a policy network (C in the WM), and therefore chooses an action purely through planning. This agent finds the best action among different predictions for different sets of actions as displayed in Figure 5 and therefore benefits directly from the model improvement to depict the consequence of its actions. The key feature of this model is the recurrent state space model (RSSM). From the current latent state (green) and a possible set of actions (light grey), the model can dream of the state of the environment and select the most rewarding one, planning and acting are therefore deeply entangled.

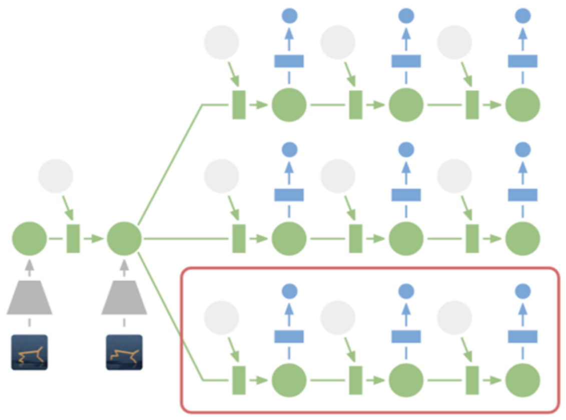


*Figure 5 Planet architecture (the input enters the encoder (grey) which is used by the RSSM (green) with the action (light grey) to plan different future and select the best one (red box) based on the rewards (blue) (Hafner et al., 2018)*

#### *2.4.2.1 The Recurrent state space mode (RSSM)*

Deterministic recurrent neural networks (RNN) can find their use in many cases by propagating information through time as in the original world model (Ha & Schmidhuber, 2018). However, they suffer from short-term memory, which displays their inability to retain information for a long time. The PlaNet is built using a more elaborate architecture, the Recurrent state space model (RSSM). The RSSM uses a gated recurrent unit (GRU) (Cho et al., 2014) to compute the deterministic recurrent state. This model is composed of two gates, the reset, and the update gate, which enable control of how much information is retained in the recurrent state. Those models are deterministic and are well known to carry forward the relevant information for many steps.
Stochastic state space models (SSM) are a probabilistic extension of RNN, it predicts distribution over latent spaces from which one potential future is sampled from this distribution. However, those models are difficult to train, due to the stochasticity of the time model, which may lead to the loss of information over time.

The combination of both (Gru & SSM) into one model (RSSM) split the state into stochastic and deterministic parts, allowing the model to robustly learn to predict multiple futures. The deterministic part allows one to remember information for a long time. The stochastic component is needed as the agent does not learn without it, because tasks are stochastic and partially observable. Partial observability refers to the concept where an agent cannot fully perceive the complete environment at a given time step. Stochasticity refers to the presence of randomness in the environment or interactions between the environment and the agent. Both are major constraints in reinforcement learning as they introduce uncertainty, increasing the task's complexity**.** This architecture is not new and can be seen as a sequential VAE.

### 2.4.3 Actor critic:

Actor-critic methods ((Sutton & Barto, 1998),Figure 6) in RL resemble the classic MPD framework with the additional component of the value function. It uses temporal difference (TD) learning approaches that incorporate a separate memory structure to represent the policy and value function independently. The actor, responsible for action selection, represents the policy, while the critic, assessing and critiquing the actor's actions represent the discounted value estimate function considering multiple steps in the future. The learning process is on-policy, requiring the critic to evaluate and critique the current policy being followed by the actor. Actor-critic methods are particularly interesting for model-based RL because they offer a way to decouple the policy and value function, allowing for more flexibility and adaptability in learning and integrating longer reward horizons. The actor-critic architecture enables efficient learning updates based on the critic's evaluation, driving improvements in the actor's decision-making and policy selection.

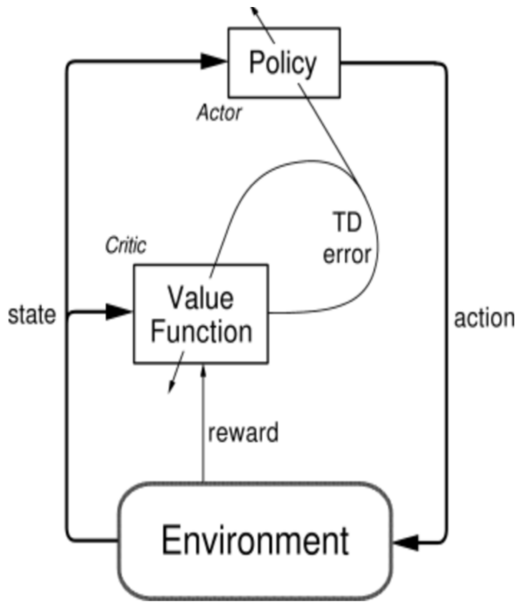


*Figure 6 Actor critic architecture (The environment outputs a state used by the policy network to choose actions, which are evaluate by value function receiving the reward and state) (Sutton & Barto, 1998)*

### 2.4.4 Dreamers:

Dreamers (Hafner et al., 2019) build upon the foundation of PlaNet (Hafner et al., 2018) while decoupling planning and acting using an actor-critic approach. The planning displayed in PlaNet is computationally expensive, decoupling planning, and acting will save computation. The dreamer use the RSSM presented in the PlaNet section with the main function to build an accurate representation of the world using latent forward dynamics. From these latent representations dreamers will learn to act using the actor model, those actions will be criticized by the reward model which will backpropagate gradients to update the policy. This enables the agent to predict how its action affects the rewards. In addition, the model-based method can be limited in how far ahead they can accurately depict the impact on their action on the state, therefore the value network estimates the sum of future rewards after the planning horizon and constrains the policies. Like the reward, the value network will backpropagate gradients to update the actor-network. These capabilities represent a pivotal shift in RL, offering agents a strong imagination that empowers them to depict the world multiple steps in the future and learn policies without interacting with the raw environment. This model limits the computational effort of planning while considering long and short-term consequences of actions.

During the last iteration of Dreamer (V2, V3), several improvements have been made to the model; the focus will be on depicting the last version, Dreamer V3.

## 3 Materials & Methods:

The materials used in this research will be described using the Markov decision process as the environment is built following this template. This description of the environment will include a general description but also all the communication space between the environment and the agent. This includes the state, reward, and continuation space. Finally, the agent (DreamerV3) will be described in more detail.

### 3.1 Simulated Environments:

The environment is an adapted version of the AgroUnity simulator (Carbone et al., 2020) of a drone flying over a sugar beet field with randomly distributed patches of defective plants visible in black (Figure 7).

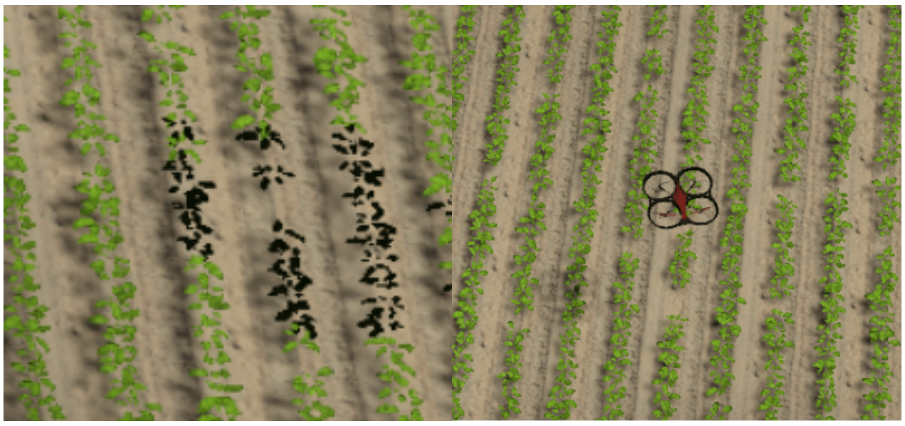


*Figure 7 Picture of the simulated environment (black plants: defective, green plants: healthy)*

Table 1 presents the main fixed settings of the field during all trials and Figure 8 provides a sketched interpretation of the field. The crop has a size of 20x30m, the area in which the drone can fly is double ($ratio_{total\ field}$) the size of the field. The drone flies at an altitude of 4m which makes the field of view 4x4 m. The row width is set to 0.75m and the plant spacing within the row is set to 0.25m, which makes a total of more than 3000 plants in the field.

*Table 1 Parameters of the environment*

| Name of parameter | Values | unit |
|---|---|---|
| Width crop | 20 | m |
| Length crop | 30 | m |
| $ratio_{total\ field}$ | 2 | - |
| $ratio_{outfield}$ | 0.5 | - |
| Altitude | 4 | m |
| Total number of plants | +/-3000 | Plants |
| Number of defect plants | Random (100,150) | Plants |
| FoV (field of view) Drone | 4*4 | m |
| Distance between rows | 0.75 | m |
| Distance between plants | 0.25 | m |
| Percentage of spawn plant | 95 | % |
| Number of Leaves per plant | Random (1,7) | Leaf/plant |

In each training episode, a random crop configuration is utilized, the randomness is integrated as the central position and shape of the defective patch at the field level. At the plant level, the randomness is included as the plant number of leaves, the plant texture from 5 textures, rotation and offset from their row to increase variability. To ensure a fair comparison, the seed remains fixed while evaluating the agent's performance. Consequently, the evaluation is consistently conducted on the same sequence of fields. The reset position for the Dreamer and the Row by Row (Figure 1) agent varies, primarily because the latter necessitates starting from a specific position. The RbR, starts in the bottom left corner of the crop denoted by the red circle in Figure 8. The dreamer will start randomly on the edge of the field, the purple line in Figure 8 when training and similarly to the field starts at a random position with a seed when evaluating.

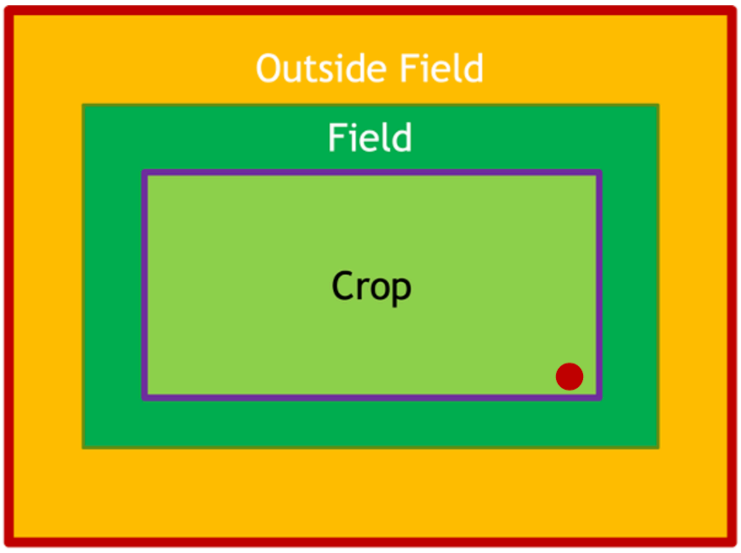


*Figure 8 Representation of the environment (The crop contains the plants, the field has no plant, the outside field has no plants and is negatively reward if visited, the boundary(red) stops the simulation, the red dot is the starting position for RbR)*

The simulation environment is implemented as Markov Decision Process (MDP, see section 3.1). Each observation represents a sequence containing the state described in section 4.1.1, the reward presented in section 4.1.2 and the continuation space in 4.1.4. The observation is then coupled to the agent's chosen action, presented in section 4.1.3.

### 3.1.1 States space:

The state space represents the input senses of the agent. It consists of a RGB image and a cover map. Data for the world model is gathered during the actor-critic's learning process. As the actor-critic refines its decision-making policy by interacting with the environment, the resulting experiences are captured and used to update the world model.

**RGB image:**

The initial state used for our experiments was a sequence of images. The environment outputs an RGB image of size 128x128 pixels from Unity from a simulated top-down view camera below the drone. It is then resized linearly to the 64x64 pixels required by the dreamer's input.

**Cover map:**

The cover map is a 64x64 matrix with two channels created to provide additional context and memory to the agent during exploration. The environment has been discretized using the complete field (Figure 8) each pixel covers an area of 0.6 m$^2$. The covered area of each pixel is smaller than the drone's field of view (FOV, Table 1 presents the main fixed settings of the field during all trials and Figure 8 provides a sketched interpretation of the field. The crop has a size of 20x30m, the area in which the drone can fly is double ($ratio_{total\ field}$) the size of the field. The drone flies at an altitude of 4m which makes the field of view 4x4 m. The row width is set to 0.75m and the plant spacing within the row is set to 0.25m, which makes a total of more than 3000 plants in the field.

Table 1), ensuring that the entire region assigned to a pixel can be observed fully when the drone is located at a given position. The first channel is the current position (Figure 9A), with the pixel value representing a binary state indicating the current position in the discretized space. The second channel is the history of the visited location (Figure 9B), the pixel values are binary. Together, the cover map and current position aim to provide the agent with a more comprehensive understanding of its environment, specifically in terms of areas already explored, thereby improving the overall performance. Both cover map channels can be made in real time without the use of prior knowledge of the positions of the defective plants in the field.

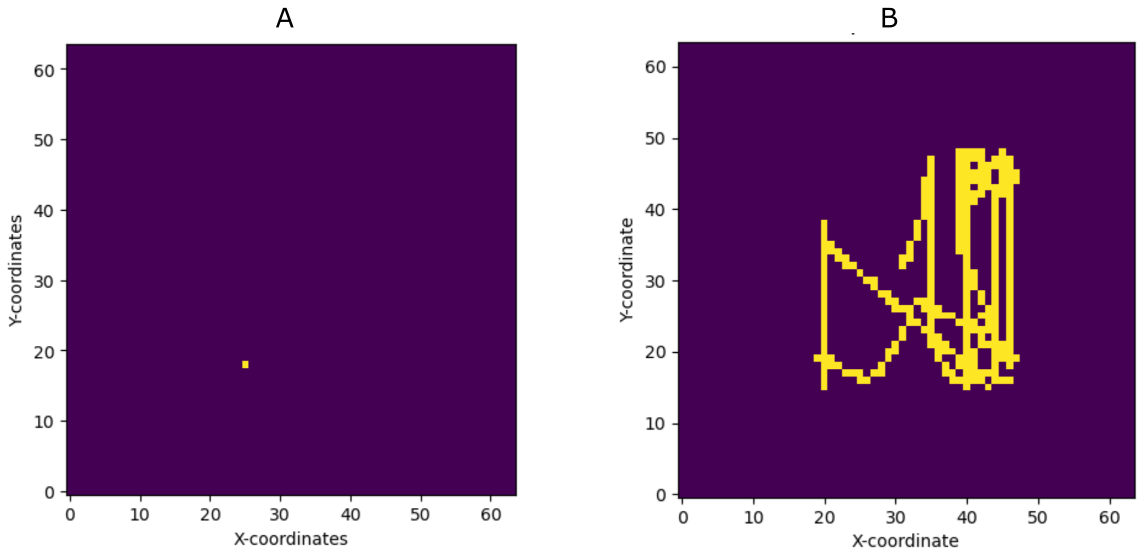


*Figure 9 Cover map containing the current position of the drone (a) and the history of visited positions (b).*

### 3.1.2 Reward space:

The reward function directs the agent in the desired direction, toward the goal of the task. The reward function is composed of one positive reward(R) and two negative rewards(C) computed at every step. The accumulation of all three represents the value function and is maximized by the agent. The coefficients are displayed in Table 2. The choice of the coefficient did not require to be balanced for the detection reward. However, the cost of actions and out-of-field exploration had to be balanced so that the agent would not terminate prematurely the task.

*Table 2 Reward coefficient*

| Name of reward | Symbol | Type | Value | units |
|---|---|---|---|---|
| Action | AC | Step | -0,2 | /step |
| Detect a defective plant | DR | Step | 1 | /plant |
| Out of field exploration | OFC | Step | -50 | /step |

The first negative reward is the cost of navigation ($C_{navigation}$) in the field by acting. It uses an immediate and continuous cost as the action is taken (AC).

$$C_{navigation} = AC \qquad [-] \qquad (1)$$

This cost aims to emphasize the need to minimize the number of steps which is the requirement of our problem.
The second one is the reward of detecting a defective plant given by:

$$R_{Detection} = \sum ND * DR \qquad [-] \qquad (2)$$

The reward is given proportionally to the newly defective (ND) plant found in the step and the coefficient of this reward (DR). The detection mechanism is triggered by the plant being in the visible area. The location of the plant is defined as the ground truth center of the plant, the seed. The visible area also named field of view (FOV) is calculated with:

$$FOV_d = 2 * (A - FL) * \tan\left(\frac{AOV_d}{2}\right) \qquad [-] \qquad (3)$$

$$AOV_d = 2 * / \arctan\left(\frac{SD_d}{2 * FL_d}\right) * \frac{180}{pi} \qquad [-] \qquad (4)$$

Where AOV is the area of view, FL is the focal length from the lens and SD is the sensor dimensions, the results are computed in both dimensions (d) of the image. At 4 m altitude (A) the FOV results are provided in Table 1, and Figure 10 provides a sketch representation.

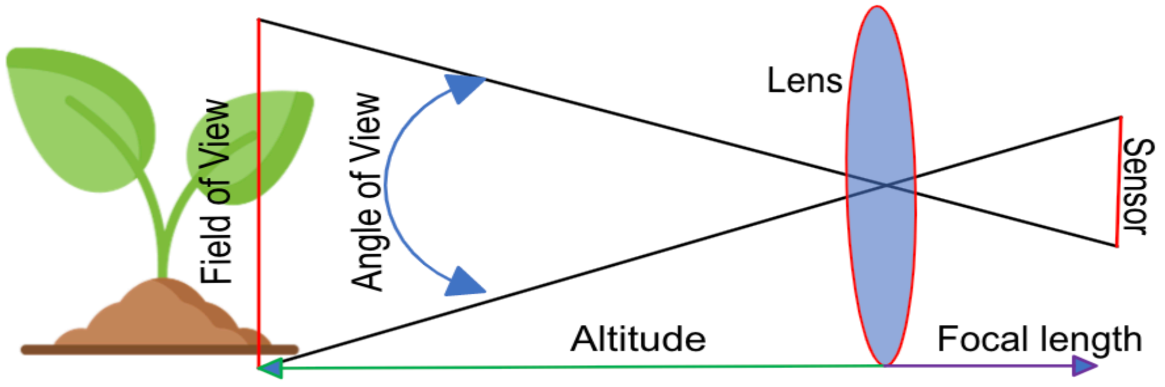


*Figure 10 Field of view of camera diagram (The observable area (FOV) within the camera's reach, influenced by the angle of view and distance to object (altitude), shaped by the sensor's dimensions and the lens' focal length)*

This detection method induces perfect detection however a portion of the plant might be in the visible area before the reward is triggered as the plant grows bigger than the seed.

The last immediate negative reward ($C_{OF}$) occurs conditionally to the exploration of the outer field is given by:

$$C_{OF} = \begin{cases} OFC, & DP_{x,y} > (ratio_{totalfield} - ratio_{limit\,field}) * CD_{x,y} \\ & 0\,, \; otherwise \end{cases}, \qquad [-] \qquad (5)$$

where DP is the drone position, the values of ratios $and\ CD_{x,y}$ (crop dimension) are given in Table 1. $Ratio_{limitfield}$ (Table 1) determines the size of the outside field (Figure 8).
The coefficient (OFC) rewarded is set big enough to dissuade the agent from exploring too far from the crop. This emphasizes the need for exploring the location with the plants while not reaching the limit of the environment and therefore terminating the simulation.

The total reward (equation 6) is the symlog linear integration of the three rewards previously presented, a trick used by dreamerV3 to deal with scales and frequencies of rewards described in section 4.2.3.

$$Total\ R = Symlog(-C_{navigation} + R_{detection} - C_{OF}\ ) \qquad [-] \qquad (6)$$

### 3.1.3 Action space:

The action space is the possible set of actions that the agent can take, classical control of drones is usually subdivided into 4 degrees of freedom, pitch, yaw, roll and height the two used are presented in Figure 11, where forward backwards is used instead of pitch.

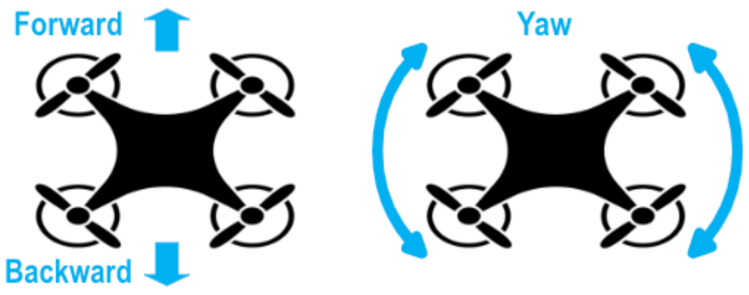


*Figure 11 Drone Control of the experiment ((forward-backwards), Yaw (rotation Left or Right))*

The task was intentionally designed to remain in 2 degrees of freedom for simplicity those are solely used for navigation using the Pitch and Yaw. The actions are discrete as the environment is designed to teleport the drone between viewpoints. Table 3 displays and summarizes the action space of the agents in the environment.

*Table 3 Action space*

| Task | Type | Action | Distance | New area |
|---|---|---|---|---|
| Navigation | Discrete | Forward/Backward | 0.5 m | 2 m2/step |
| | | Yaw (L/R) | 15° | 1.6 m2/step |

The forward/backward motion was intended as the exploration where the rotation would be used to set the direction of the exploration and not to explore the field. The information gained from rotating should therefore be minimal compared to the information gained when moving backwards and forward. The yaw was set to an angle that is not 90° to allow diagonal movements over the field. The rotation angle (Table 3) was set to allow a wide range of motion to the agent and use the flexibility of the model to explore as much as possible in a continuous space. Moreover, it was made sure that the newly discovered portion from the rotation was smaller when travelling forward.

### 3.1.4 Continuation state:

Three distinct stopping criteria have been established to regulate the agent's behavior. The primary criterion involves the achievement of the task's objective, specifically, the identification of all defective plants. This criterion holds a significant influence over the navigation cost. By locating the plants fast the agent should be able to limit the cumulative cost of exploring the field.

The second and third criteria are designed to regulate the exploration of the agent within its environment. The second criterion triggers cessation when the agent reaches the predetermined theoretical limit of the environment, as indicated by the red line in Figure 8. The addition of this stopping criterion was the reason for adding a high negative reward to the agent when exploring the outfield. As the agent could decide to shortcut the task and terminate the simulation by crashing in the border.

Lastly, a third stopping criterion dictates that if the agent surpasses 800 steps, the simulation will be terminated. This measure ensures that the agent's exploration remains within a defined temporal boundary, preventing prolonged simulations that might lead to suboptimal actions and therefore exploration. As well as leaving enough time to complete the task as it is double the required steps for the RbR.

## 3.2 DreamerV3 agent:

The key architecture of the Dreamers has been displayed in the background section; in this section, we will review the core of the Dreamers and we will elaborate on the variation found in the newest version of Dreamers. To summarize what has previously been explained, Dreamer architecture is subdivided in two, the world model (Hafner et al., 2023a) uses the RSSM to learn compact representations of sensory inputs through autoencoding and enables planning by predicting future representations and rewards for potential actions in a certain state. The second part is the actor-critic model which will learn how to behave in an environment by interacting with the latent dynamics of the world model by predicting the reward, value, and action. The first two will constrain the third one in the long-term desired direction. Figure 12 & Figure 13 displays the model architecture and training process. The world model and the actor-critic do not share gradients, and during behavioral learning, the gradients of the world model are frozen.

### 3.2.1 World model:

The world model architecture consists of an image encoder coupled to the RSSM to predict the image, reward, and continuation flag its architecture is depicted in Figure 12. The first network used in the world model is the encoder which uses a Convolutional Neural Networks (CNN) to extract features from the inputs ($x_t$). This encoder is the key to getting the latent state on which latent forward dynamics model work and consists of the first part of the VAE. It is then coupled to a Recurrent state space model (RSSM) architecture which relies on 3 networks:

**Representation model:** The representation model is an MLP using the embedding of the encoder and the deterministic recurrent state($h_t$) to predict the distribution ($q_\emptyset$) over the stochastic latent state ($z_t$). One latent representation is sampled ($\sim$) from the vector of SoftMax distributions as displayed in equation 7.

$$Z_t \sim q_\emptyset(Z_t \mid h_t, x_t) \qquad [-] \qquad (7)$$

**Dynamic predictor:** The dynamic network uses an (MLP) to predict probability distribution (p) over the latent state ($\hat{z}$) from the recurrent state ($h_t$), without access to the current input. One latent representation is sampled (~) from the vector of SoftMax distributions as displayed in equation 8.

$$\hat{z} \sim p(\hat{z} \mid h) \qquad [-] \qquad (8)$$

As this network depicts the scene with sole access to the recurrent state it is, therefore, the key to the dreaming capabilities of the world model. This model is crucial for planning as it will have to encode information from the recurrent state and the action for multiple steps.

**Recurrent model**: The recurrent model uses the GRU to predict the next deterministic representation (($h_t$), equation 10) using the previous recurrent state($h_{t-1}$,), latent representation ($z_{t-1}$) and action ($a_{t-1}$).

$$h_t = f_{\emptyset}(h_{t-1}, z_{t-1}, a_{t-1}) \qquad [-] \qquad (9)$$

The RSSM is trained by minimizing the Kullback-Leibler (KL) divergence between the dynamic predictor ($p(z_t \mid h_t)$) and the next stochastic representation ($q_{\emptyset}(Z_t \mid h_t, x_t)$) as displayed in equations 10 & 11.

$$L_{dyn}(\phi) = max\left(1, KL\left[sg\left(q_{\emptyset}(z_t \mid h_t, x_t)\right) \parallel p_{\emptyset}(z_t \mid h_t)\right]\right) \qquad [-] \qquad (10)$$

$$L_{rep}(\phi) = max\left(1, KL\left[q_{\emptyset}(z_t \mid h_t, x_t) \parallel sg\left(p_{\emptyset}(z_t \mid h_t)\right)\right]\right) \qquad [-] \qquad (11)$$

KL Divergence is used to measure the distance between probability distributions. The difference in these losses is the stooping gradient operator(sg). Stopping the gradient aims to cancel the propagation of the error, assuming the value is constant. Each loss therefore trains the representation model (equation 10) and the dynamic predictor (equation 11). This matching enables the dynamic predictor to learn to predict the latent representation without access to the current input.

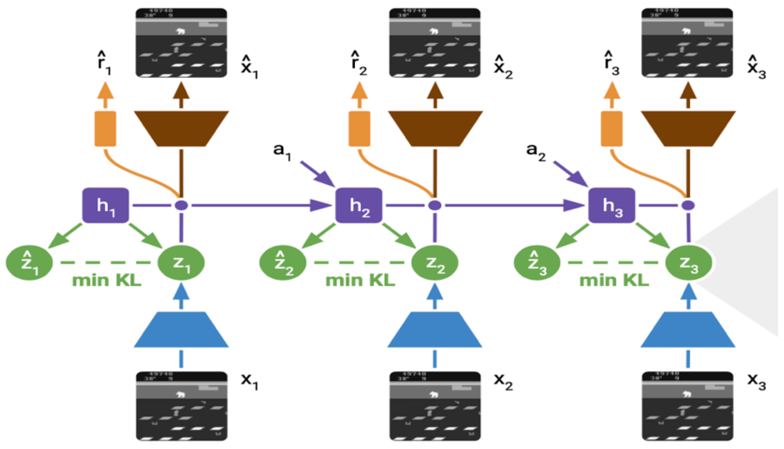


*Figure 12 The world model architecture (The input (X) is encoded (bleu), the RSSM (green) uses the representation network to generate stochastic state(Z)with the recurrent state and encoded input. The dynamic network generates the deterministic latent state ($\hat{Z}$) from the recurrent state, then the GRU network propagates in time the recurrent(h) and latent(z) state with the action (a). From these states the reward ($\hat{r}$)) is predicted (orange), finally, the image input is reconstructed ($\hat{X}$) by the decoder (brown)*

**Decoder**: The decoder is a transposed CNN doing the opposite of the encoder, it decodes the latent state (z) and recurrent state (h) into an input format ($\hat{x}$) understandable to humans by estimating the probability (p) over the reconstructed input from which one is sampled (~) as depicted in equation 12.

$$\hat{x} \sim p(\hat{x} \mid h, z) \qquad [-] \qquad (12)$$

It consists of the last part of the VAE. The decode is trained by minimizing the symlog (Figure 14) loss, this loss is the mean square error of the transformed value (equation13).

$$L(\theta) = \frac{1}{2}\left(f(x, \theta) - symlog(y)\right)^2 \qquad [-] \qquad (13)$$

This image is not expected to be the exact representation of the original image as it lost information during the compressing phase.

**Reward predictor ()**: The reward network uses an MLP to predict the probability distribution (p) of the reward from the recurrent (h) and latent state (z). One latent representation is sampled (~) from the vector of SoftMax distributions as displayed in equation 14.

$$\hat{r} \sim p(\hat{r} \mid h, z) \qquad [-] \qquad (14)$$

It uses the same discrete regression approach as the critic network depicted below. This model will have to minimize the symlog loss.

**Continuation predictor**: The continuer uses an MLP to predict the binary episode continuation flags ($\hat{c}$) of the action from the recurrent (h) and latent state (z). One latent representation is sampled (~) from the vector of SoftMax distributions as displayed in equation 15.

$$\hat{c} \sim p(\hat{c} \mid h, z) \qquad [-] \qquad (15)$$

This model is trained by minimizing the binary classification loss visible in equation 16.

$$L_{bin} = abs(\hat{c} - c) \qquad [-] \qquad (16)$$

### 3.2.2 Actor-Critic Learning:

The actor-critic is based on the actor-critic approach (Figure 6) however learns behaviors exclusively by the extracted features of the world model; its training process is depicted in Figure 13.

**Actor-network**: The actor-network is an MLP that learns a policy ($\pi_t$), the probability of over the action space to sample (~) an action ($a_t$) that maximizes the expected returns considering the current state ($s_t$):

$$a_t \sim \pi_t(a_t \mid s_t) \qquad [-] \qquad (17)$$

The actor will balance exploration (finding new solutions) and exploitation (using best policies) through an entropy regularizer (E). This regularizer introduces a balancing mechanism, ensuring the actor-networks actions consider both exploration (finding new solutions) and exploitation (using the best policies). The scale and frequency of rewards were shown to be significant impactors leading to smaller rewards ending up losing their impact (Haarnoja et al., 2018). To deal with the scale without affecting the frequencies of the return, large returns are scaled down without scaling up small ones by dividing them by their scale only for returns bigger than 1. It allows on the one hand to have only one entropy scale for both sparse and dense rewards. On the other hand, it enhances exploration in situations with sparse rewards while maintaining high performance with dense rewards in a heavily randomized environment.

**Value network:** The Value network is an MLP that aims to predict the probability distribution of the expected sum of discounted rewards ($v_t$) after the planning horizon using discrete regression knowing the current state ($s_t$). The return is transformed using the symlog function detailed below and then two-hot encoded to speed up training of a possibly widespread return. Two hot encodings is a generalization of one hot encoding for continuous values as its values can fall between buckets ($b_i$).

$$v_t = \mathrm{E} * p(\, b_i \mid s_t \,) * b_i \qquad [-] \qquad (18)$$

This discrete regression has shown to be a key improvement specifically with sparsely rewarding environments. Moreover, as the environment is stochastic rewards will not always be the same, which also justifies its performance. To train this model a categorical cross-entropy loss was used.

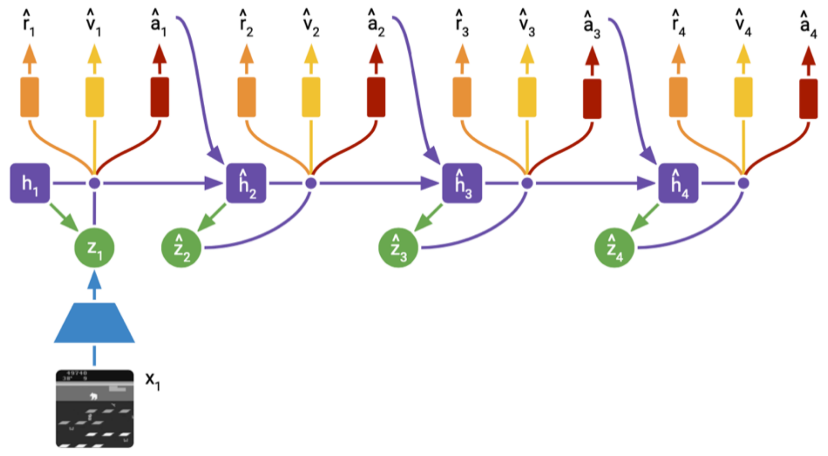


*Figure 13 Actor critic architecture with dreamerV3 (The context input (x) is encoded (Blue), the RSSM will then generate the latent state (Z) from the recurrent state and input, it can then predict the reward ($\hat{r}$), value ($\hat{v}$)and action ($\hat{a}$). From the recurrent state, the next latent state ($\hat{Z}$) is computed and propagated by the recurrent model(purple) and used to predict ($\hat{r}$, $\hat{a}$, $\hat{v}$))*

### 3.2.3 The general purpose of DreamerV3

One of the main claims of DreamerV3 is that it is a general-purpose algorithm. A general-purpose algorithm is an RL algorithm that aims to perform any task with fixed hyperparameters, obliviating the need for optimization and human intervention (Hafner et al., 2023b). Developing such a model has long been the goal of reinforcement learning. This model needs to master continuous and discrete actions, high and low-dimensional inputs, different reward scales & frequencies, and 2D and 3D environments. To achieve this goal two main tricks were used, the symlog rescaling and the use of the latent dynamics model described here.

#### *3.2.3.1 Scaling using Symlog*

The challenge in reproducing and predicting rewards and values lies in the varying scales across domains. Traditional loss functions like squared loss may lead to divergence with large targets, while absolute and Huber losses can stagnate learning (Hafner et al., 2023b). Normalizing targets using running statistics introduces non-stationarity in optimization. To address this, dreamer uses symlog predictions, a solution where a neural network learns to predict a transformed version of its targets, a bi-symmetric logarithmic function, which allows backpropagation. The symlog function (Figure 14) compresses large values, preserves signs, and approximates the identity around the origin. This simple method has shown to be a significant improvement in performing a wide range of tasks without needing to fine-tune the hyperparameters. This method was used in the decoder, the reward predictor, and the critic.

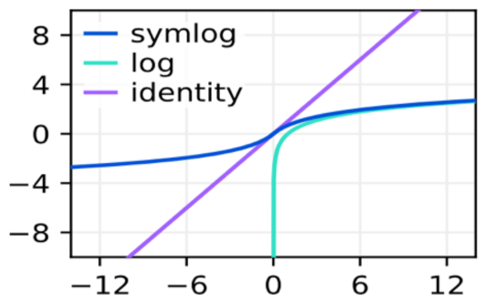


*Figure 14 Symlog scaling compared to indetity and log (Hafner et al., 2023)*

#### *3.2.3.2 Latent dynamics and model size*

In order deal with the dimensions of the input, the model encodes the input in a latent representation (z). This allows to extract key features from the inputs and represents the world in a constant latent state shape. This extraction allows the agent to learn from a latent representation which speeds up the learning process. Dreamer V3 also displays that the performance of the model, and the sample efficiency followed the size of the model. This is hot topic in deep learning as large language models have sometimes been scaled up to improve their performance, however, it has been shown that the size of the model is not always a determining factor in the performance of the model (Touvron et al., 2023).

### 3.2.4 Implementation:

The Deep RL model utilized in this research is DreamerV3 using a Pytorch implementation (NM512, 2023) replicating the original model by Hafner et al. (Hafner et al., 2023). Consistency is maintained by using the same hyperparameters across all tasks. Training persists until convergence, identified by a flattened loss. The model was trained on a computer with a Ryzen 9 5950X processor and an NVIDIA RTX 3090 for an average of 96 hours for each experiment, ensuring efficient learning and stabilization of performance.

### 3.2.5 Hyperparameters:

This agent will use the same hyperparameters (Table 4) during all the trials, those hyperparameters are the same as the original experiment (Hafner et al., 2023).

*Table 4 List and description of hyperparameters used*

| Name | Value | Definition |
|---|---|---|
| Learning rate actor critic | 3e-5 | The learning rate is a hyperparameter that determines the step size at each iteration while moving toward a minimum of the loss function during training. In this context, it specifically applies to the actor and value networks. |
| Learning rate: World Model | 1e-4 | Like the learning rate for the actor and values, this is the step size used for updating the world model during training. |
| Actor entropy scale | 3·10−4 | This parameter controls the balance of exploration and exploitation during training. It scales the entropy term in the objective function, regulating the stochasticity of the policy generated by the actor. Higher entropy encourages more exploration, making the policy more diverse in selecting actions. |
| Lambda return | 0.95 | Lambda return is a parameter used in the calculation of returns. It represents the discount factor for future rewards. A value of 0.95 means that future rewards are exponentially decayed by 5% at each time step. |
| Batch size | 64 | Batch size is the number of training samples utilized in one iteration. A batch size of 64 means that 64 samples from the dataset are processed before the model's parameters are updated. |
| Imagination horizon | 15 | It refers to the number of steps the agent looks ahead when simulating probable future states to make decisions. |
| Number of latent | 32 | The number of latent is the number of states the agent is planning on. |
| Number of categories | 32 | The number of categories refers to the number of features the world model is extracting from the input. The size of the latent state is 32. |

## 3.3 Performance metrics:

### 3.3.1 Feature extraction of Dreamer V3:

This section describes the performance metrics to assess the efficiency of the world model in DreamerV3. The overall performance evaluation encompasses the entire world model, comprising the Variational Autoencoder (VAE) responsible for encoding contextual information into latent representations and the recurrent model that propagates these representations into the future knowing the action. This is done using closed-loop, open-loop and phenotypic evaluation.

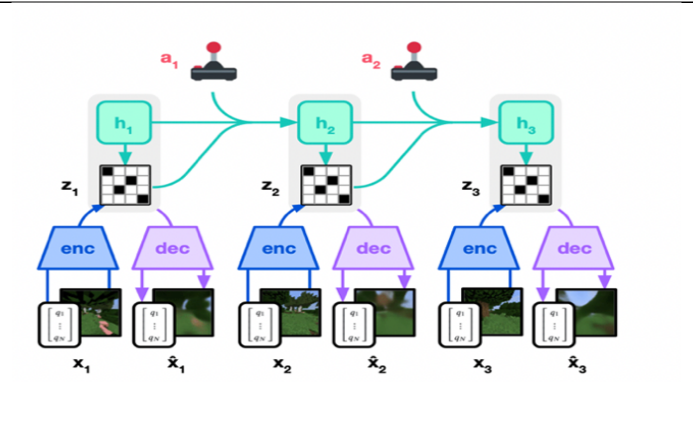
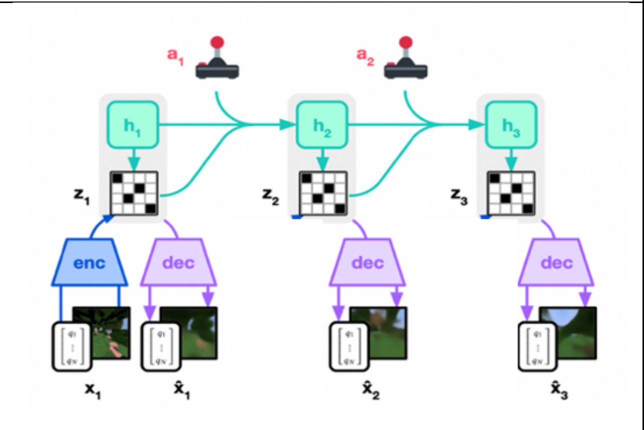
| A | B |
|---|---|
|  |  |

*Figure 15 A: Close loop evaluation B: Open loop evaluation (Hafner et al., 2023)*

**Closed-loop Evaluation:** To gauge the effectiveness of the world model performance in depicting the state with the context, we employ closed-loop error measurements. This involves comparing the reconstructed image ($\hat{x}$), obtained through a forward pass in the encoder-decoder pathway, with the original image(x). The assessment of reconstructed images encompasses both qualitative analysis of the image and quantitative evaluation using mean square error (MSE, equation 19) at the image levels and the error at the pixel level. Figure 15A summarizes the evaluation step taken in a closed loop.

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(x_i - \hat{x}_i) \qquad [-] \qquad (19)$$

**Open-loop Evaluation:** The performance of the recurrent model of the RSSM is evaluated through open-loop predictions, where a context image is presented for five-time steps alongside the corresponding action. The model then extrapolates by dreaming the situation into a latent state, therefore this evaluation focuses on evaluating the ability to learn the dynamics of the environment. Like the closed-loop evaluation, the assessment of reconstructed images encompasses both qualitative analysis and quantitative evaluation at the pixel level using the error and at the image level using MSE. Figure 15B summarizes the evaluation step taken here in three steps. Additionally, our interest extends to understanding error propagation over multiple time steps following the context presentation. Therefore, we will assess the MSE over the sequence and therefore assess the impact of the accumulation of errors while dreaming.

**Phenotypic Evaluation: In** the assessment of the model's performance on specific features, we employ a phenotypic evaluation approach. This evaluation focuses on comparing masked ground truth images with the corresponding masked reconstructed images. In a closed loop, both healthy and defective plants undergo evaluation using data from the last evaluation trial. The mask, an RGB image generated by the environment from the visible region, consists of three categories saturating the channels: healthy plants, defective plants, and soil. A binary mask is extracted from these categories and used to identify the region of interest in the images. The evaluation process utilizes a single frame to assess the image both qualitatively and quantitatively. Quantitative analysis includes measuring the mean square error (MSE, equation 19) and Euclidean distances (equation 20) at the pixel level for all color channels (red: r, green: g, blue: b).

$$Euclidian\ distance = \sqrt{(r-\hat{r})^2 + (g-\hat{g})^2 + \left(b-\hat{b}\right)^2} \qquad [-] \qquad (20)$$

This ensures the quantification of prediction errors, providing a comprehensive understanding of colored error scales on a unified level and detailed insights into the model's accuracy in reproducing specific features. Additionally, the mean square error of the entire path from the last evaluation is computed to assess the overall performance across the training. This comprehensive evaluation approach ensures a thorough analysis of the model's ability to accurately capture and reproduce features of interest during the training phase.

### 3.3.2 Behavioral comparison:

To evaluate quantitatively the performance of the agents on successfully achieving the task we will measure its task effectiveness using the detection rate oversteps. We will always compare the performance to the baseline path row by row, the performance of multiple runs will be evaluated and therefore we will assess the trends and the standard deviation.

### 3.3.3 Baseline comparison:

The row-by-row (RBR) baseline comparison was established to achieve complete field coverage with 100% accuracy, although its path is suboptimal due to encompassing regions without defective plants and extending beyond the field boundaries. To define the optimal RBR path, multiple sequences of actions were devised and iteratively repeated based on factors such as field size, step size, and field of view (FOV), one of these back-and-forth sequences is illustrated in Figure 16. In evaluating the RBR path, a random field was employed as the performance is independent of the defect distribution, relying solely on field size and FOV. The assessment involved 30 iterations to smooth the results. The anticipated performance is expected to exhibit linearity under the assumption of a uniform defect distribution, while a Gaussian distribution, resulting from an epicenter in the environment, should showcase steps within a single episode, with the average linearizing over multiple iterations. Despite its fixed and suboptimal nature for single-patch scenarios, the RBR method is anticipated to demonstrate increased efficiency as more patches are introduced into the field.

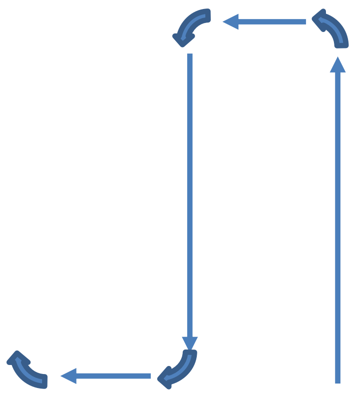


*Figure 16 Single Row by Row pattern sequence (straight arrow: forward, curved arrow: rotations)*

## 3.4 Experiments:

The main goal of this research study is to evaluate the performance of dreamerV3 for defective plant search in the case of varying, the distribution of the plants and input. The evaluation of DreamerV3 is structured into two distinct training phases corresponding to the reason for the specific use of this agent. The initial phase focuses on world model performance to extract features and plan the impact of its action on this environment. It is then followed by actor-critic training performing the goal-directed task of finding the defective plants. This structure serves as the base of our methodology. An evaluation step is conducted at 20,000-step intervals, to ensure a thorough understanding of the agent's capabilities at various stages of learning. From the evaluation steps the best-performing agent based on the minimal number of steps will be selected for evaluation.

The experimental design within this thesis aims to scrutinize DreamerV3's performance under varying conditions by manipulating two critical parameters: the input information and the patch distribution in the field. Each parameter undergoes two variations, resulting in four distinct scenarios (Table 5), alongside a baseline scenario for comparison. The following sections will delve into the specific details of each experiment, presenting a comprehensive description of their goals following the research questions.

*Table 5 List of evaluated scenarios*

| | Input | Number of patches |
|---|---|---|
| Base line 1 | | 1 |
| Base line 2 | | 5 |
| Scenario 1.1 | RGB image | 1 |
| Scenario 1.2 | RGB image | 5 |
| Scenario 2.1 | RGB image + Cover map | 1 |
| Scenario 2.2 | RGB image + Cover map | 5 |

### 3.4.1 Number of patches

The initial experience aims to address the first research question, which assesses the impact of the number of patches on the performance of the agent learning from an RGB image. To do so we will compare both scenarios 1.1 and 1.2 to address if the strategy displays different performance.

#### *3.4.1.1 Feature extraction*

Assessing the performance of the world model is crucial to address whether the performance of the actor-critic model is affected by the poor performance of the world model in depicting the scene. To assess the performance of the world model we will evaluate the three different metrics presented in the performance assessment section above. Those are the close loop errors evaluating the performance of the world model with input context, and the open loop evaluating the world model without context input, both from an independent random sample sequence of input states. Finally, the phenotypic error will be evaluated, the masked prediction error of both the healthy plants and defective plants in an open loop using the final evaluation run.

#### *3.4.1.2 Actor critic*

After the evaluation of the feature extraction model, the goal is to evaluate the agent's ability to perform the desired tasks on these extracted features. The average detected rate oversteps of 10 evaluation trials will be evaluated to address the effectiveness of the agent. A visual evaluation of the drone path will be made using a complete and uncomplete run, the completion is based on finding all defective plants. Initially, both dreamer performances will be compared to the baseline (RbR) and then between the performance of the dreamers with different numbers of patches.

### 3.4.2 Increase context

These research questions will focus on comparing the DreamerV3 performance using solely the image and with increased context by providing the cover map to the agent. This should increase the ability of the drone to locate itself in the field and could be seen as a new sense of the agent, a form of global map.

#### *3.4.2.1 Feature extraction*

In the feature extraction research question, we will assess the same performance as in the previous section for the image. However, we will also assess it for the cover map in the open and closed loop prediction error using the same approach presented in the previous feature extraction section.

#### *3.4.2.2 Actor critic*

For this experiment, we will address the same performance criterion as described in the performance assessment. Here we will compare the performance of both dreamers with the different inputs as well as compare the performance of scenarios 2.1 and 2.2 with each other and the baseline RbR.

# 4 Results

## 4.1 Number of patches:

### 4.1.1 Feature extraction

The closed-loop evaluation aims to assess how the model can extract features of the scene and reconstruct them from the context input. From the images presented in Figure 17, we can visually assess the performance of the world model in accurately depicting the scene. Overall, the images within the closed loop exhibit globally high level of reconstruction quality, sharing a consistent and qualitative resemblance. Distinct observations emerge in close loop video prediction (Figure 17): Firstly, the orientation of rows, plant types, and shadows in the field is well-preserved and accurately positioned. Secondly, a notable characteristic is the loss of details in the plants, resulting in the representation of a continuous and blurred row. Individual plant properties and the missing plant in rows become less discernible. Examining the phenotypic error (Figure 20), it becomes evident that plants located outside the main patch tend not to be adequately encoded, often being replaced by healthy plants. At the pixel level, there is a noticeable discrepancy in error, with defective plants exhibiting a higher error in the rows compared to their healthy counterparts. The error at the pixel level of healthy plants is almost not visible.

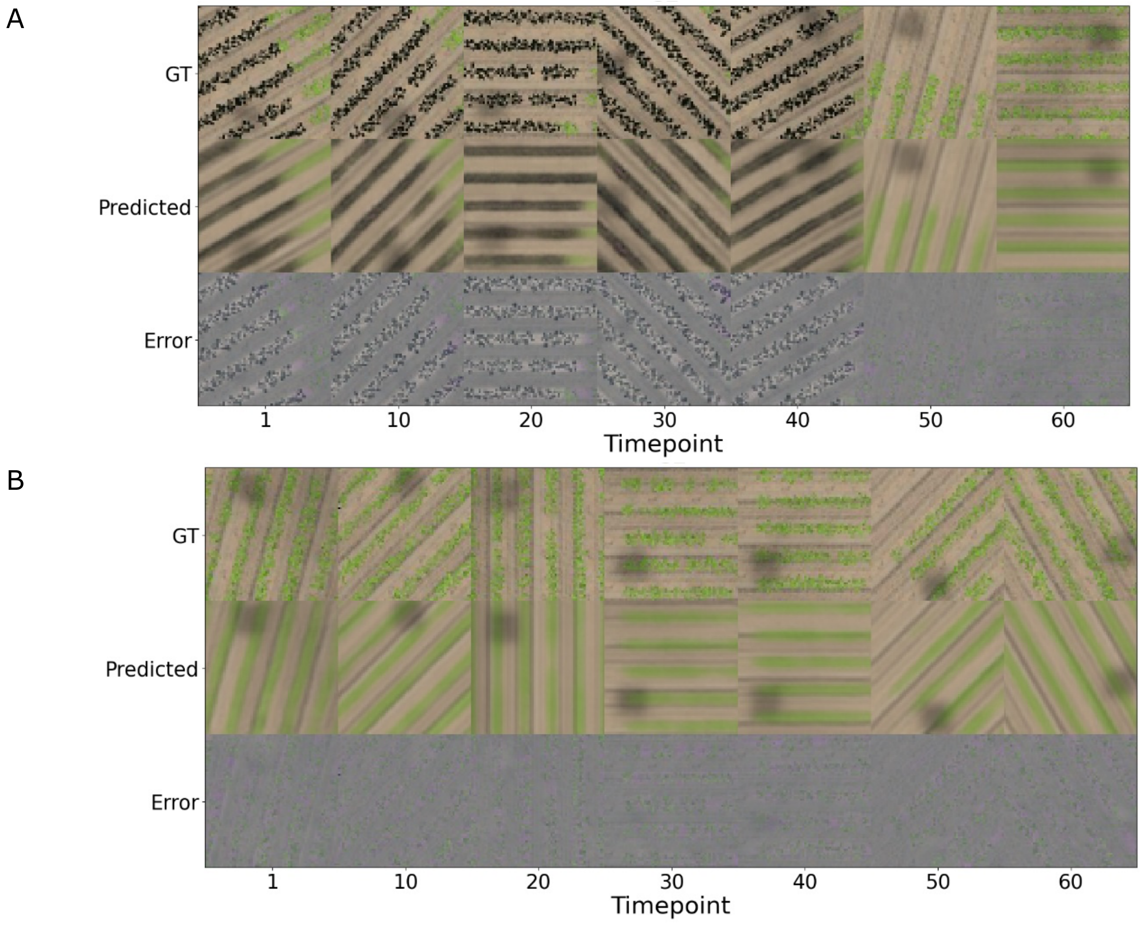


*Figure 17 Close loop video prediction scenario 1 (A: Single patch, B: multi-patch, GT: true image, predicted: world model prediction, Error: difference (GT, predicted))*

The open loop assessment focused on evaluating the world model’s ability to make predictions without initial context, learning the environment's dynamics, both scenarios display remarkably similar performance. In examining the open loop images in Figure 18, the reconstruction displays a similar pattern to the closed loop however additional patterns can be identified. The rows and the shadow remain well encoded as they are maintained along the entire close-loop prediction. In the final stage (40/50/60), the model displayed a tendency to continue lines with healthy plants. At the pixel level, the error does not seem to build up significantly with a relatively constant error.

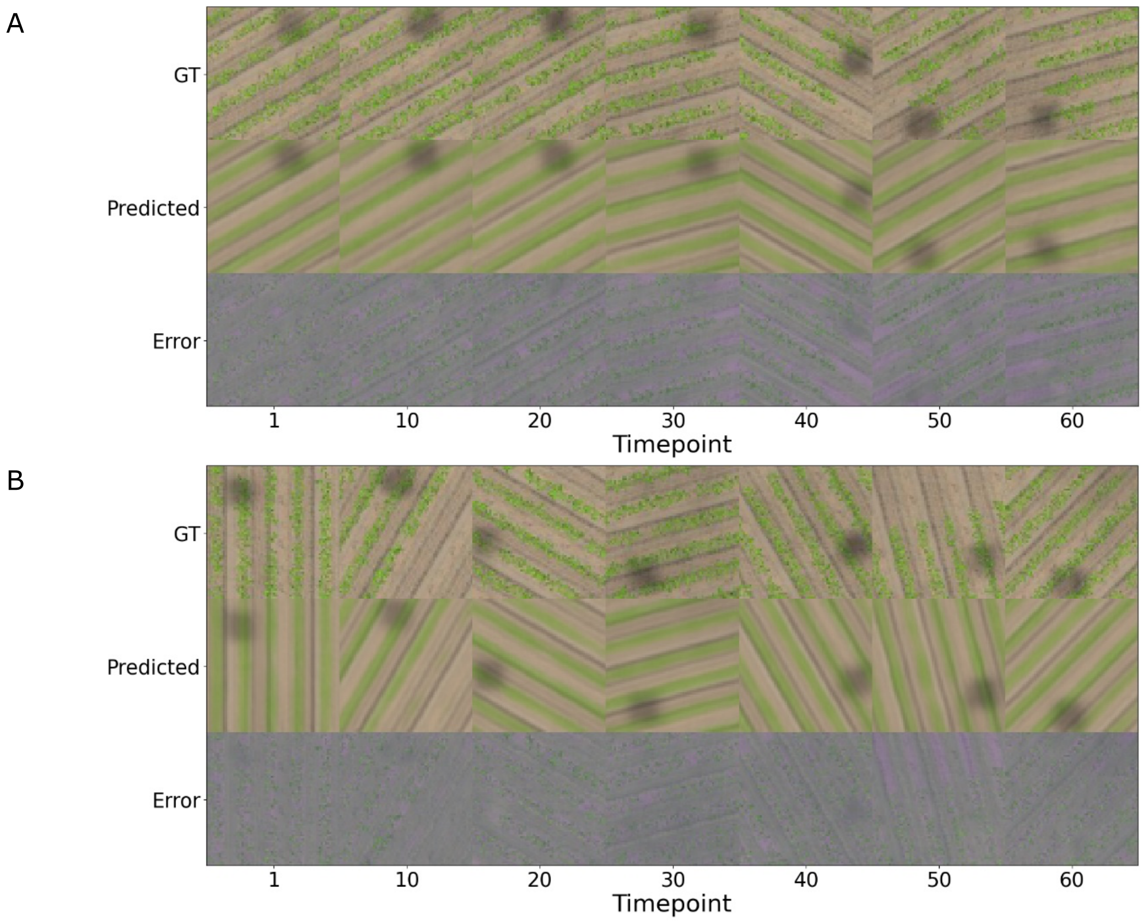


*Figure 18 Open loop video prediction scenario 1 (A: Single patch, B: multi-patch, GT: true image, predicted: world model prediction, Error: difference (GT, predicted))*

The evaluation oversteps (Figure 19) of the close loop (CL) prediction seems to display a common decreasing trend for the first steps of the loop followed by a relatively stable performance. The magnitude of the error between both scenarios differs significantly before step 50, which is coherent with the high pixel error related previously, caused by defective plants. The end of the single-patch scenario resembles the general trend of the 5-patch scenario where both are reconstructing a healthy portion of the crop.

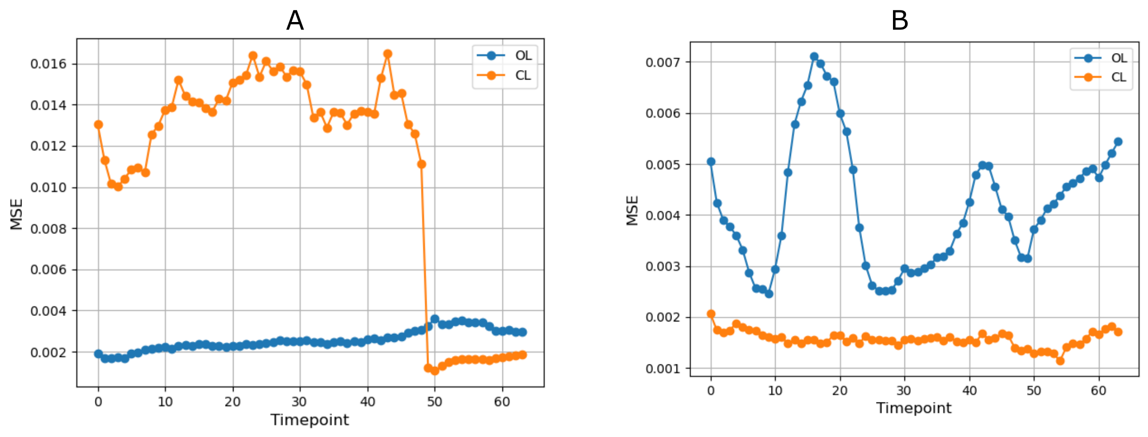


*Figure 19 Image error oversteps scenario 1, Close Loop (CL) and Open Loop (OL) for the single patch (A) and the multi-patch (B)*

The evaluation of the error across steps (Figure 19) provided insights into error dynamics within the model. Notably, both closed-loop MSEs showed a consistent decrease during the initial five close-loop steps as in the closed-loop evaluation, reflecting the positive influence of the recurrency of the model. After the context steps it shifts to a more ascending trend where the error of the predictions builds up. In the five patches scenario, we find a non-steady trend which might be the result of the highly predictable field due to its geometrical aspect. In summary, the open loop assessment revealed the world model's proficiency in feature extraction but also highlighted its tendency to persist in certain predictive patterns which might be caused by a lack of global awareness. The observed dynamics in MSE emphasized the critical role of context in sustaining accurate predictions over successive steps.

Understanding the phenotypic error is crucial for decoding how the world model encapsulates features of interest. This error, operating within a closed loop, reflects patterns inherent in the loop itself presented earlier. The novelty lies in the examination of specific features, as illustrated in Figure 20, encapsulating the core elements of this methodology. Analyzing the mask, it displays a reasonable quality however some pixels are not attributed to the crop, especially visible in the healthy mask. The masked Ground Truth (GT) is wider than the crop and a portion of the ground is always visible in between rows. Even though some healthy leaves are visible in the masked GT of the defective plants, those regions are also visible in the defective mask as overlapping visible in yellow. The GT image looks of better quality for the healthy plants than for the defective plants. Figure 20B also displays that defective plants are replaced by healthy plants, a trait which was not encountered in the close loop but is a pattern of this reconstruction. Figure 20 succinctly portrays the reproduction error between defective and healthy plants evident using both MSE and Euclidean distance. A notable difference is visible as defective plants are reconstructed less accurately than their healthy counterparts noted by higher average pixel values. The graphical representation (Figure 21) accentuates the performance differences in the world model, emphasizing higher error rates for defective plants at each evaluation step. The world model displays a steep decline of the error for both scenario and plant state followed by an almost flattened error. The MSE of defective plants in the single patch scenario has displayed an unstable behavior with a form of binary trend in the flattened section. In summary, the phenotypic error analysis provides valuable insights into the world model's performance in encoding features related to both healthy and defective plants. The results have been consistent throughout both scenarios, the results of the multi-patch are visible in Appendix 2.

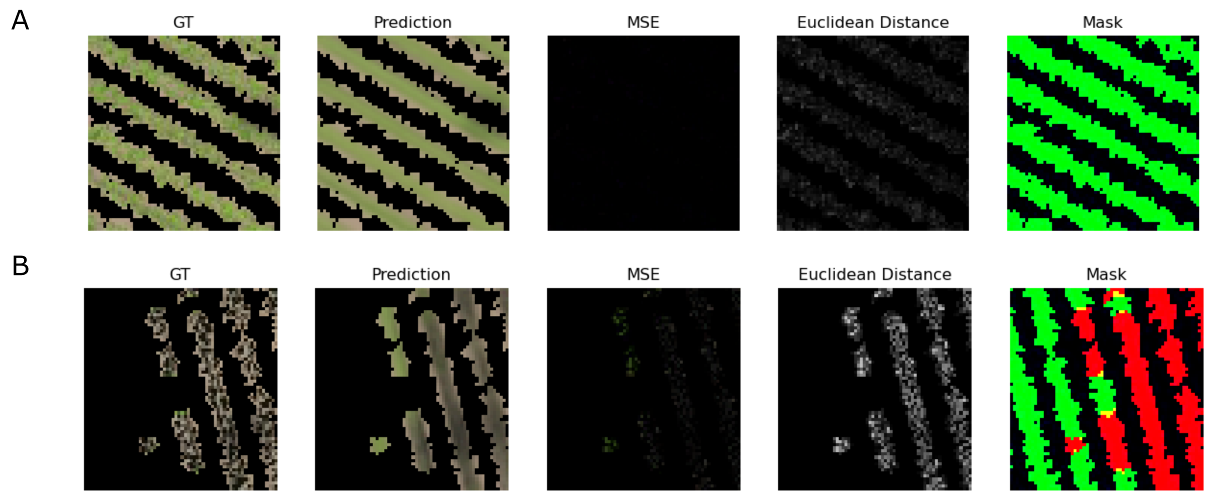


*Figure 20 Phenotypic error scenario 1.1 (A: healthy plants, B: defective plants, the original image (GT) and the predicted image (Prediction) are compared in the MSE and Euclidian distance and finally the mask used is visible (Mask)*

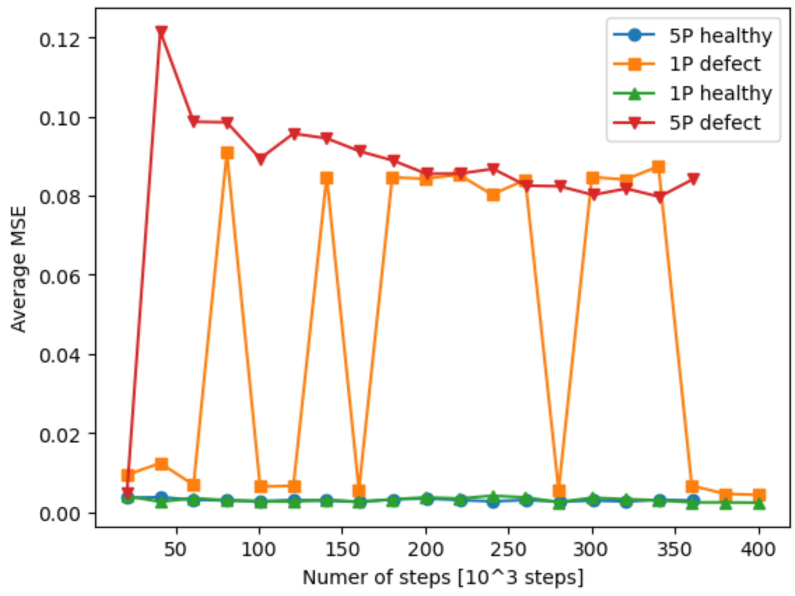


*Figure 21 Average phenotypic MSE at evaluation for scenario 1 (1P = Single patch, 5P: multi-Patch)*

### 4.1.2 Actor critic:

The second focus of this research question aims to assess the performances of the agent to perform the desired task with different numbers of patches. In all experiments, the performance of the row-by-row agent after 30 trials has been averaged and displayed in Figure 22**.** All defective plants were found after 400 steps in the environment with a steep linear detection ratio, the paths can be visible in Figure 23. We find a large standard deviation due to the distribution of the defective plants. As the patches are bigger in the single-patch scenario, the detection steps are less frequent but bigger which explains the high standard deviation.

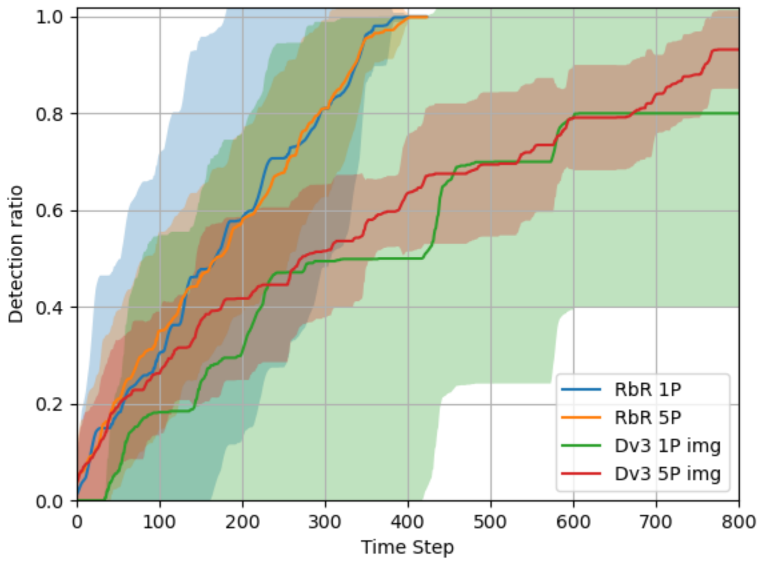


*Figure 22 Average detection overstep with standard deviation scenario 1 for Row by Row (RbR) and DreamerV3(Dv3) for single (1P) and multi-patch (5P).*

The dreamerV3 demonstrates a rapid initial detection, with both scenarios exhibiting a logarithmic progression characterized by a sharp increase in detection at the start, which gradually levels off. Dreamer did however not outcompete the RbR even in the initial phase. As the drone is exploring the remaining becomes harder to detect, moreover, the leftovers require revisiting the epicenter of the patch which increases the length of the exploration. In the single patch scenario, the trend is very staircases, with Plato followed by a steep increase. The plateau at 0.5 just after 300 steps with a high standard deviation suggests that half of the trials have beaten the row by row. Similarly, the final 200 steps are completely flat with a high standard deviation, which suggests that two of the ten trials did not find any defect. One of these incomplete paths is visible in Figure 23A, which demonstrates a high overlap of the path with significant action being rotation or looping on itself which is suboptimal. The performance of the agent in the multi-patch scenario displays a similar detection trend with a smaller standard deviation.

Both muti-patch and single-patch path (Figure 23) appeared to have learned to stay in the crop region and perform active perception to some extent. However, they seem to struggle to explore efficiently and retain where they already have explored the field denoted by the highly overlapping paths particularly predominant in the uncompleted single patch (Figure 23A). Visibly both scenarios do not fall in the same type of error, in the multi-patch scenario (Figure 23B) the agent failed to perform active perception for this patch.

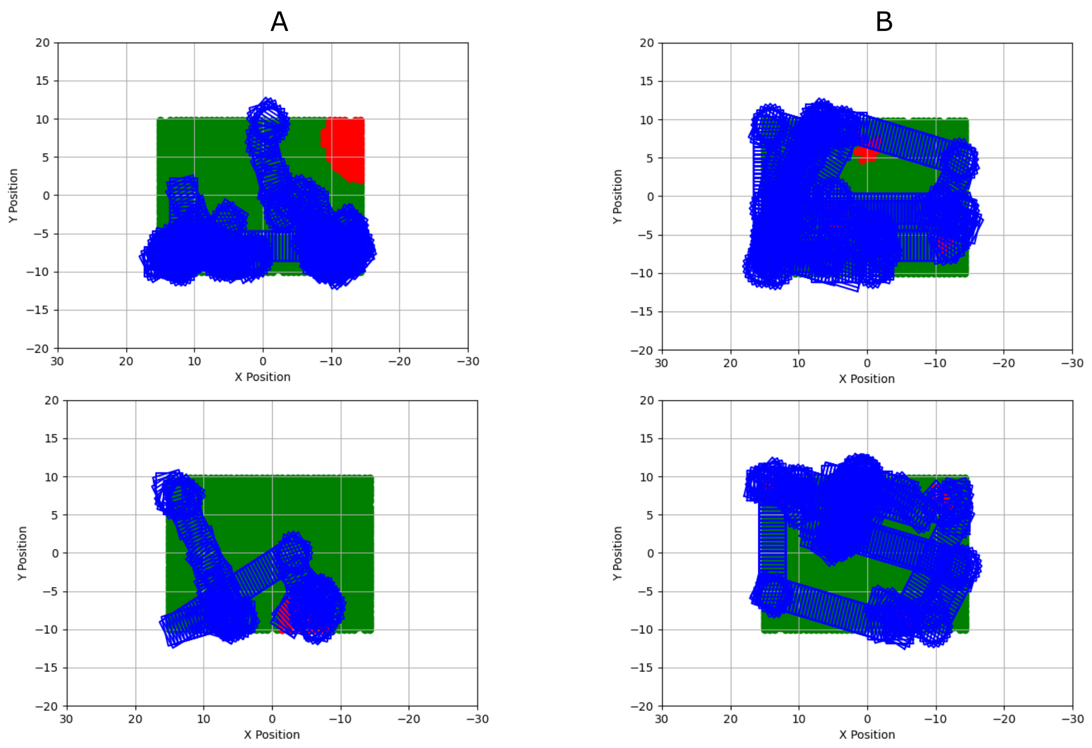


*Figure 23 Example of drone path scenarios 1.1 (A) and 1.2(B) uncompleted(top) and completed(down) (healthy plants in Green, defective plants in red and field of view in bleu)*

## 4.2 Increase context

### 4.2.1 Feature extraction

The result for the close loop prediction of the image displays similar behavior to the previous research question depicting the performance of the world model. The single-patch scenario has a sharp decrease at the beginning followed by a slower but globally decreasing error. In the five-patch scenario despite the initial improvement of reconstruction, the close loop error has a two-level performance which we already encountered before arising from the presence of defective plants as visible in images of Figure 25B. The reproduced context image (t=1) of Figure 25 displays a qualitatively less good representation than the other ones with increased blur. We here also see the impact of defective plants outside the main patch one plant is absent in steps 1 and step 20. Even more for the two plants in the same row, the world model seems to have fused them in step 20 even though there is a healthy plant in between.

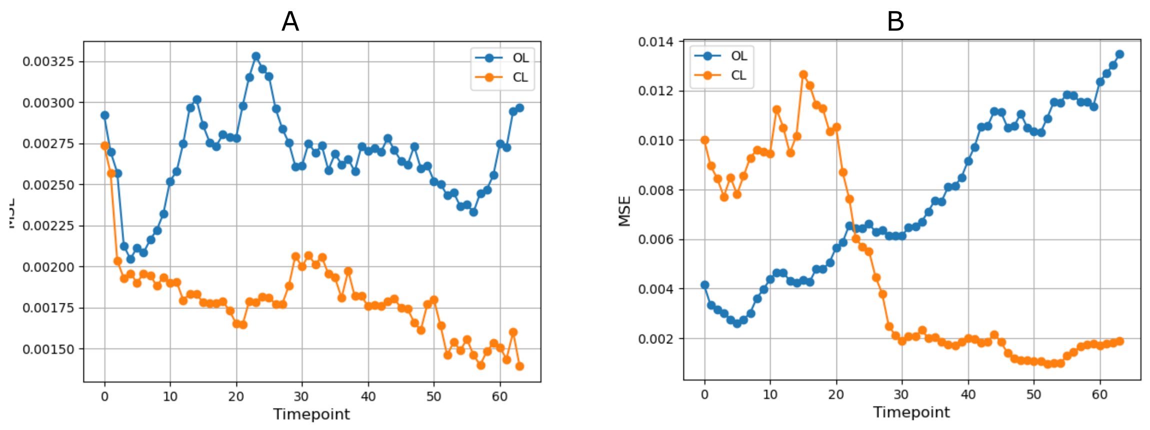
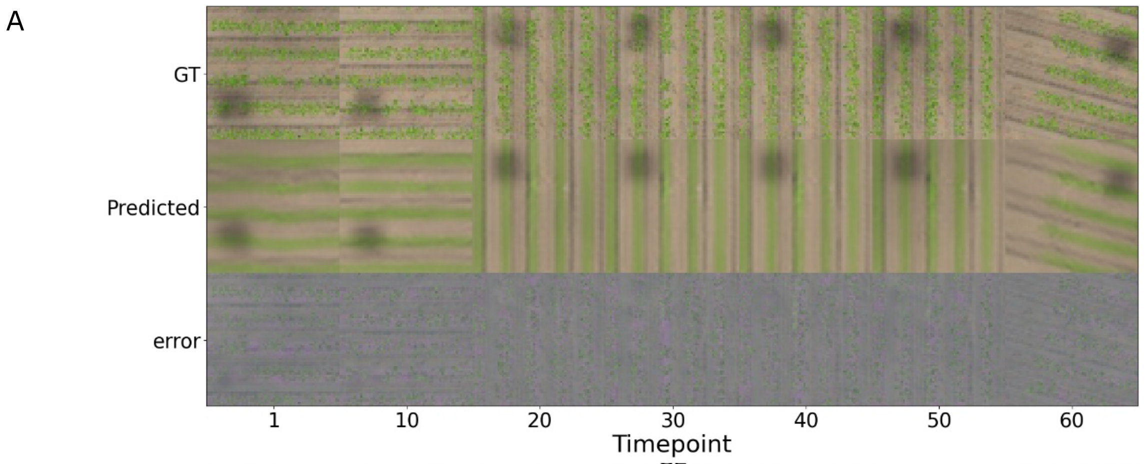


*Figure 24 Image error oversteps scenario 2, Close Loop (CL) and Open Loop (OL) for the single patch (A) and the multi-patch (B)*



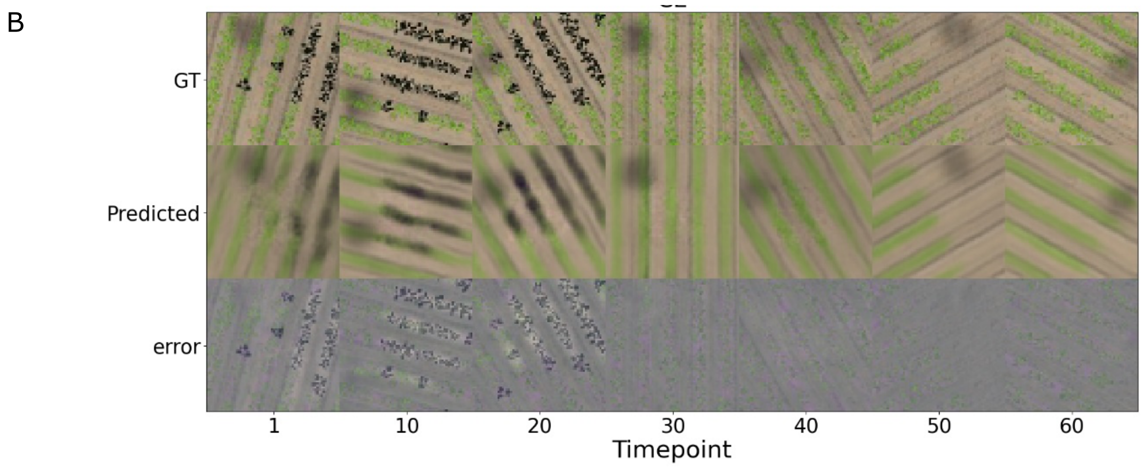


*Figure 25 Close loop video prediction scenario 2 (A: Single patch, B: multi-patch, GT: true image, predicted: world model prediction, Error: difference (GT, predicted))*

In the open loop both scenarios display a similar reproduction of the image to the previous depiction of the world model performance in an open loop. The single-patch video prediction showed a similar initial behavior with a V shape then the trend unstably flattened as already encountered in scenario 1.2 however the magnitude is lower. From the image, we can also see that the field is covered with healthy plants which might contribute to this flattening curve. We see that the field is well reproduced, and the rows stay present for many steps.

The 5-patch scenario displays a similar trend to the one already presented, with an initial decrease followed by a relatively steady increase. In Figure 24B we can find one steeper increasing phase from time step 20 to 40, with the visual evaluation of timesteps in Figure 26B, we see an increasing presence of defective plants, which the agent cannot be aware of as it has never seen them resulting in predicting healthy plants. This absence of prediction of a defective plant can be seen as the same magnitude of error as the correctly predicted defective plants.

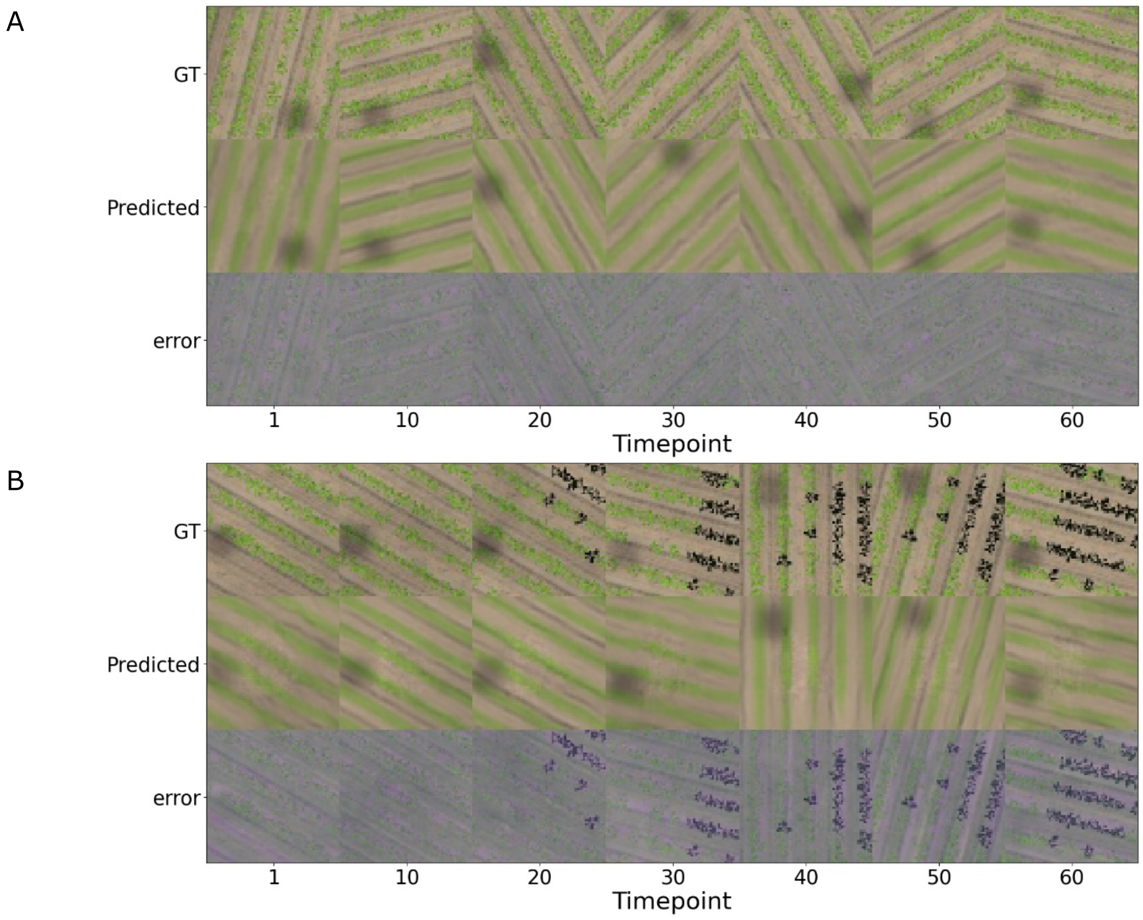


*Figure 26 Open loop video prediction scenario 2 (A: Single patch, B: multi-patch, GT: true image, predicted: world model prediction, Error: difference (GT, predicted))*

The Phenotypic Error for the second experiment (Figure 27,appendix 3) displays a similar result compared to the one from the previous scenarios using solely the RGB image. The mask seems of the same quality, the masked grand truth (GT) carries a portion of the soil, and the defective masked grand truth seems of the worst quality compared to the healthy one. The errors suggest similar performance at the pixel level with higher error and Euclidean distance for defective plants as encountered earlier. The mean error at each evaluation step (Figure 28) also demonstrates that the reproduction error is in general higher for defective plants than for healthy ones. Comparable results have been found in both distributions.

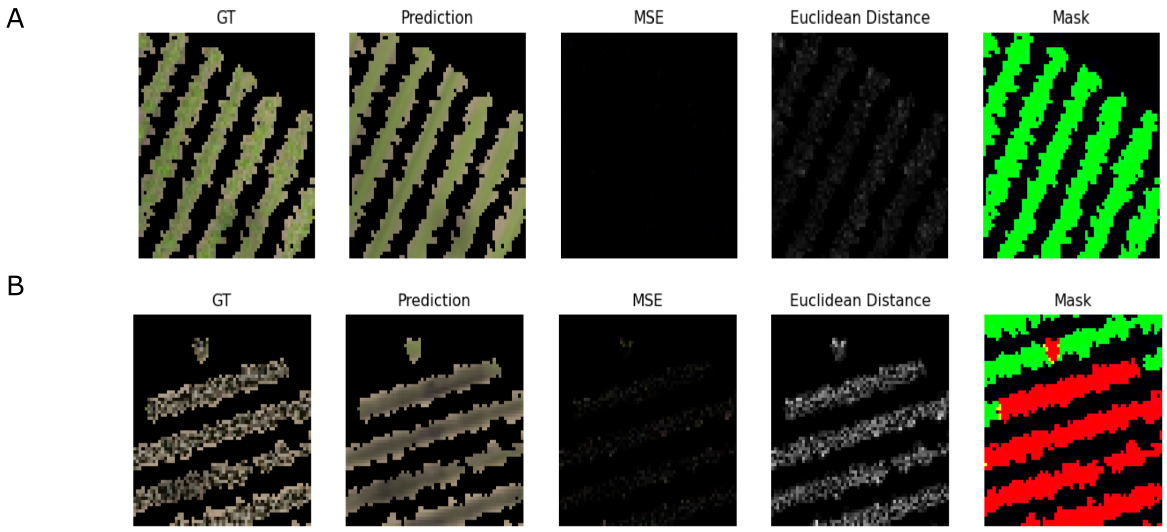


*Figure 27 Phenotypic error scenario 2.1 (A: healthy plants, B: defective plants, the original image (GT) and the predicted image (Prediction) are compared in the MSE and Euclidian distance and finally the mask used is visible (Mask)*

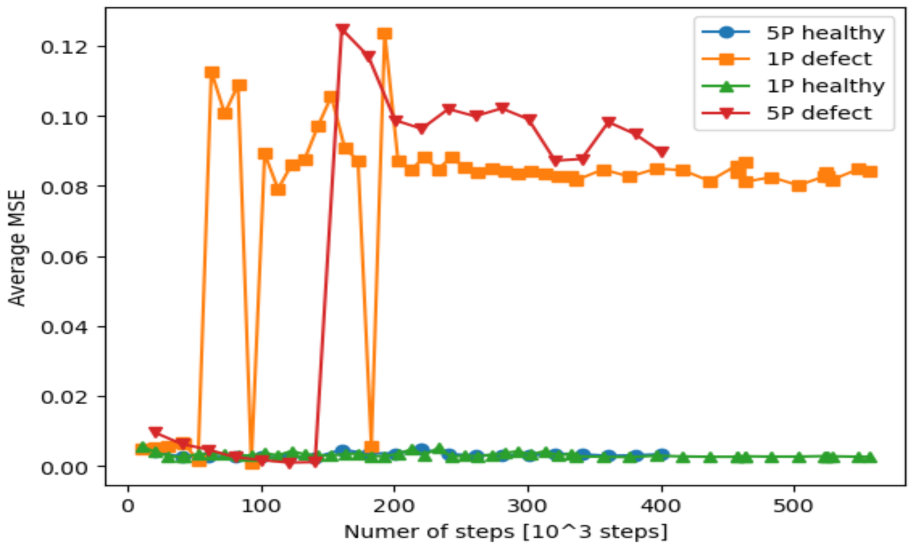


*Figure 28 Average phenotypic MSE at evaluation step scenario 2 (1P = Single patch, 5P: multi-Patch)*

The close loop cover map (Figure 29) displays reasonable reconstruction patterns, however not precise, as visible in the error of Figure 29. The path is roughly reconstructed and widened in most cases. We again find a loss of details in the reconstruction of the input. Similarly, to the image, a form of blurred reconstruction is made especially visible in Figure 29B.

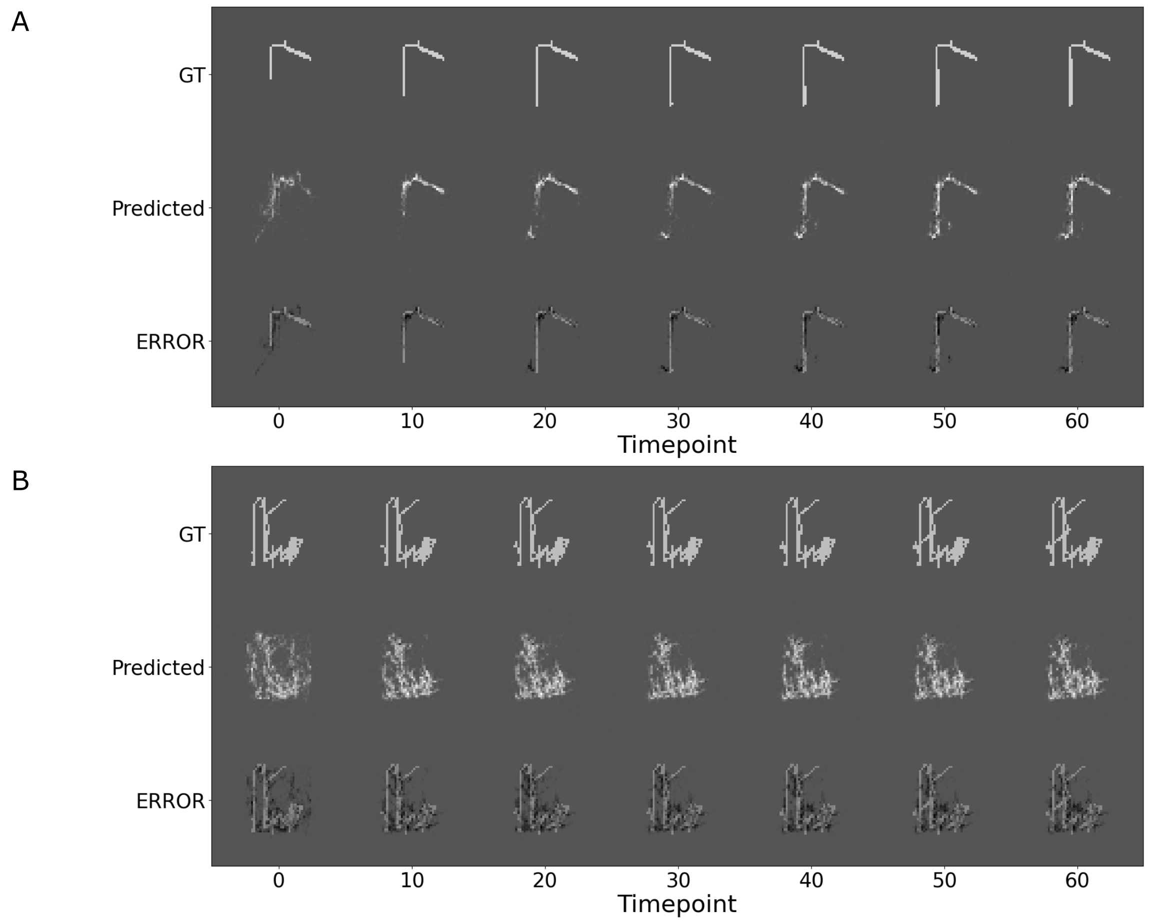


*Figure 29 Close loop prediction cover map (A: Single patch, B: multi-patch, GT: true cover map, predicted: world model prediction, Error: difference (GT, predicted))*

The reproduction of the current position is invisible depicted in Figure 30.

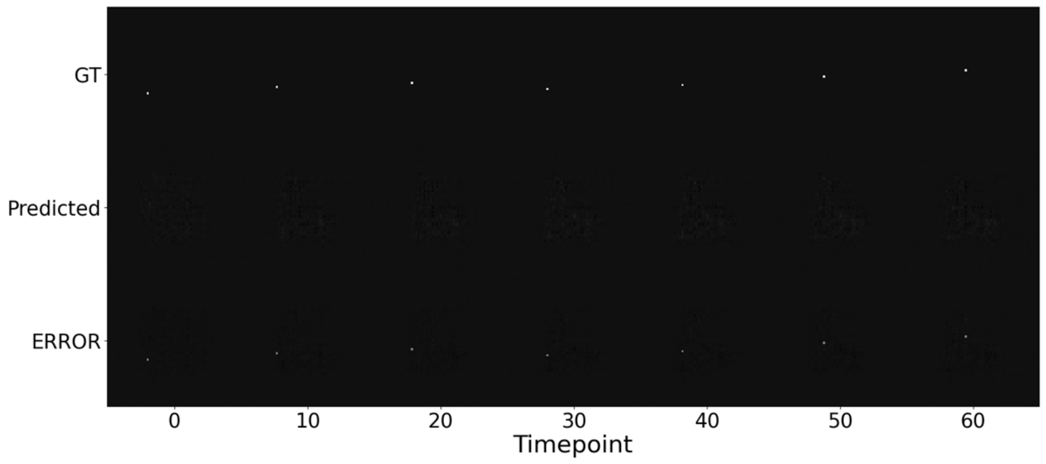


*Figure 30 Close loop video prediction of the current position (GT: true cover map, predicted: world model prediction, Error: difference (GT, predicted))*

The error oversteps (Figure 31) display a similar trend to the image close loop error for both defect distributions, with an initial decrease followed by a relatively steady performance. The magnitude of the error is however bigger for the 5-patch scenario. However, it is visible that the path is longer therefore after normalizing the error by the number of visited pixels, we found that the magnitude difference is cancelled as visible in Appendix 2.

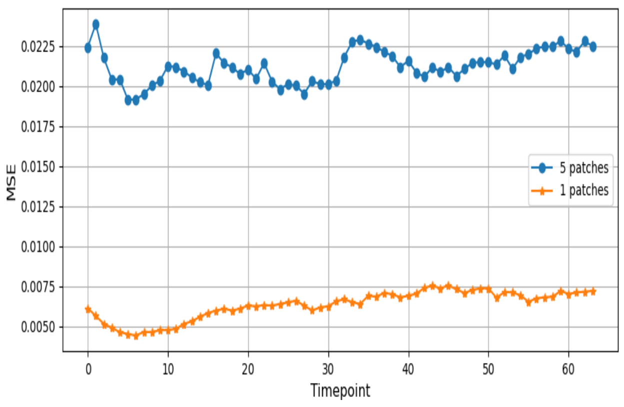


*Figure 31 Cover map MSE oversteps for close loop*

In the single-patch open loop scenario (Figure 32), the cover map gradually fades as the path is constructed, indicating two noteworthy observations. Firstly, it suggests that the Dreamer struggles to retain its cover map for an extended duration. Additionally, the agent faces challenges in effectively representing its presence in the field, as it fails to depict its movement accurately. Contrastingly, in the 5-patch scenario, a distinct trend emerges. The world model exhibits the ability to maintain the path of the cover map over time. However, a notable limitation surfaces, as the model fails to update the map with newly visited areas. As a result, the steps after the context of the scenario appear similar, lacking the necessary distinction.

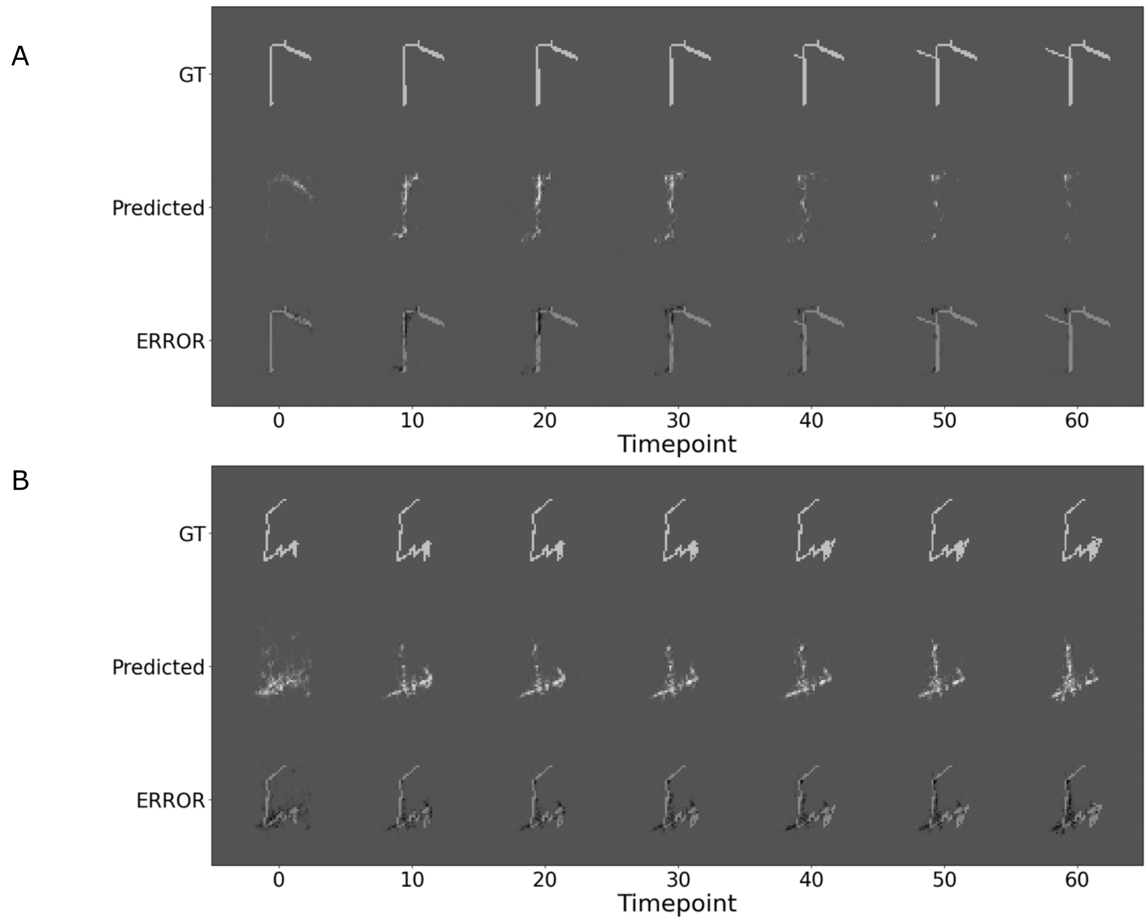


*Figure 32 Open loop map prediction (A: Single patch, B: Multi-patch, GT: true cover map, predicted: world model prediction, Error: difference (GT, predicted))*

Examining the open loop error oversteps, a consistent ascending trend is observed. This behavior mirrors a pattern previously encountered, featuring a sharp initial decrease in error followed by a gradual buildup of errors. This trend suggests the need for further investigation into the model's learning dynamics and potential adjustments to enhance its error management over time.

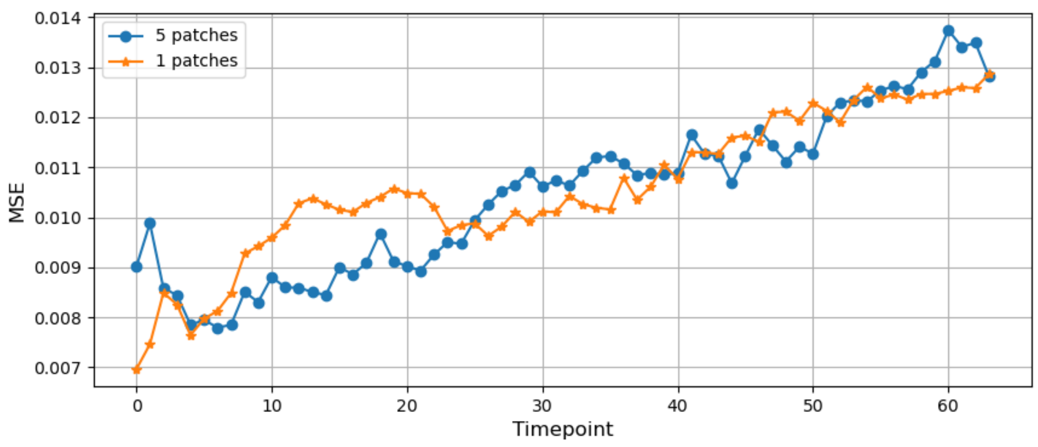


*Figure 33 Cover map MSE oversteps for open loop*

### 4.2.2 Actor critic

The Dreamer in the single-patch environment equipped with the cover map displays significant improvement in performance and almost outcompetes the row-by-row method visible in green in Figure 34. The advantage of the dreamer is particularly visible in the initial steps of the exploration, outcompeting the RBR for more than 200 steps. The result of the dreamer displaying a high standard deviation suggests that most trials (9/10) were already finished before the 400 steps. We see the Plato at 0.9 which is the sign of just a single run that is not finished. The overlap seems reduced in the presentation of the drone's path in Figure 35A. More optimized strategies have been learned to explore rapidly a significant portion of the field.

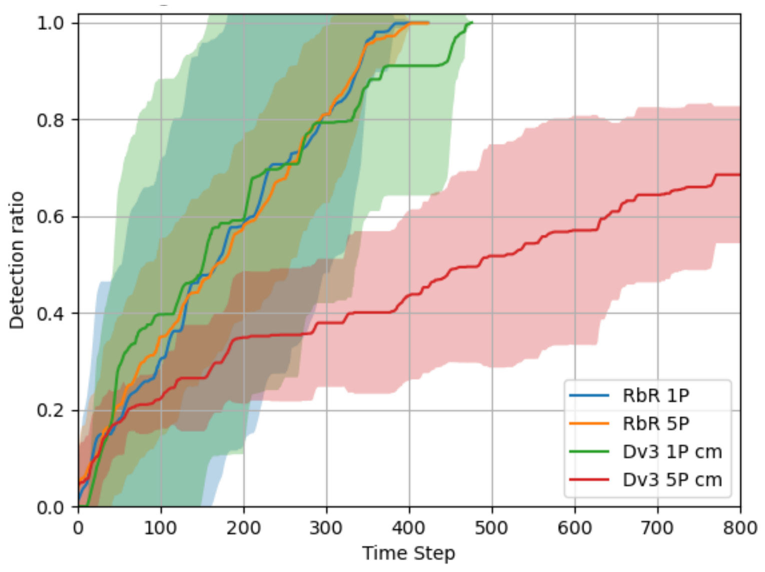


*Figure 34 Average detection overstep with standard deviation scenario 2 for Row by Row (RbR) and DreamerV3(Dv3) for single (1P) and multi-patch (5P).*

In the scenario with multiple patches, we find that the performance did not improve with the addition of the cover map. The performance displays a logarithmic detection of the plants. The significant overlap displayed in Figure 35B demonstrates that the exploration strategy is suboptimal.  The impact of the number of patches has also been evaluated in this section, we can see that the best performance with 5 patches is significantly worse than the 1 patch.

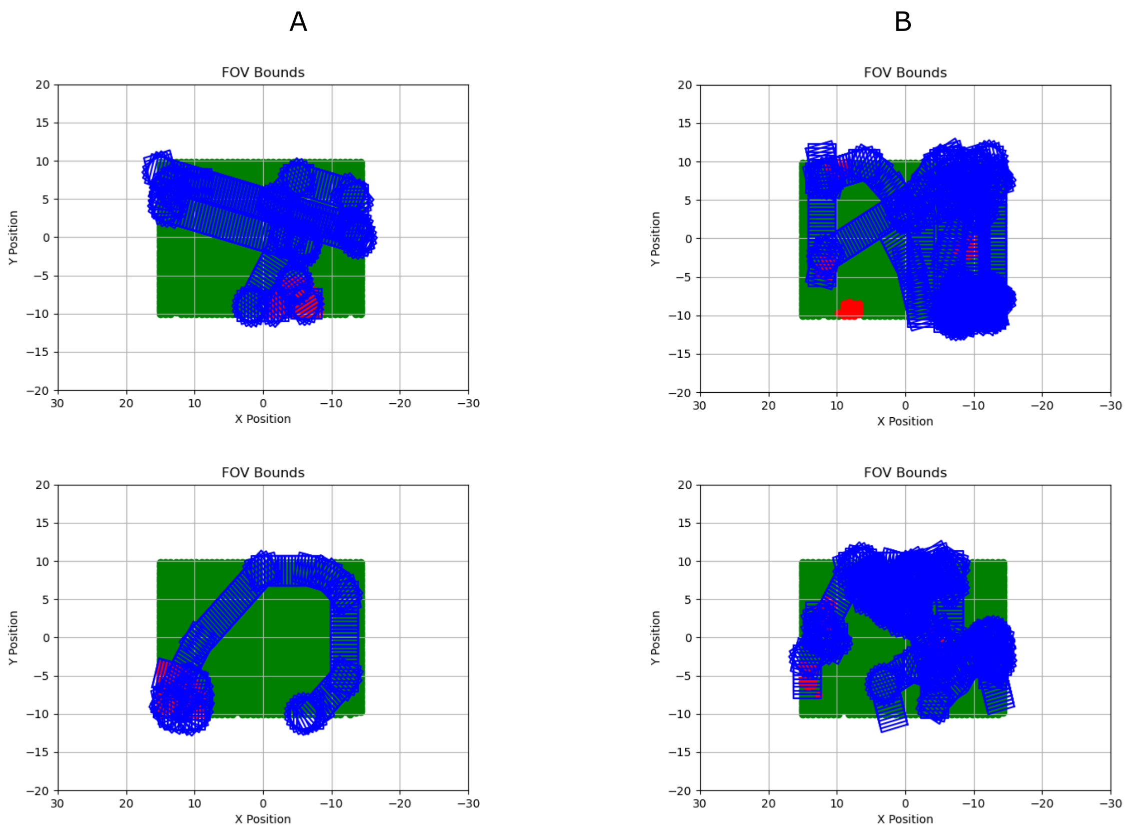


*Figure 35 Drone path scenarios 2.1 (A) and 2.2(B) uncompleted or longest(top) and completed(down) (healthy plants in Green, defective plants in red and field of view in bleu)*

# 5 Discussion:

This study has demonstrated that the dreamer can be a powerful agent to abstract the world using the image. It displays consistent results through the different scenarios with a biased performance toward healthy plants. This reconstruction seems enough to perform the task with the reconstruction of the rows, the shadows, type of plant. However, a loss of detail was visible, with an averaging of the plant's detail and loss of larger features such as missing plants or healthy plants outside main patches. For the history of the visited position, the model displays a similar performance to the image with a rough reconstruction however the current position was poorly reconstructed. On the other hand, the agent was able to find defects in a crop with high flexibility regarding the initial conditions while not consistently completing the task with a highly overlapping path when solely using the image. It has however displayed the ability to perform active perception deviating from the trajectory to find defective plants. The multipath has shown a good exploration strategy however lacked balancing the exploration and active perception. The agent with global awareness was well-equipped to search for a single patch, with a flexible starting position which was not the case in the multi-patch scenario and nearly outcompeted the baseline comparison.

## 5.1 World Model

The world model consistently exhibits accurate performance in abstracting globally the inputs, as evident in the open loop. The anticipated similarity in patterns across these outcomes stems from the consistency of inputs and the completion of the training process. However, a noteworthy limitation surfaces concerning the model's precision, noticeable in both the generated images and cover maps. The inherent characteristic of encoding states into a compact representation contributes to a generalized loss of details such as leaves or offsets of plants which are in our case not detrimental. However, the loss of bigger details such as missing plants raises concerns about the precision of the encoding process. The discrepancy in encoding accuracy, especially in capturing fine details, prompts the need for further investigation and refinement of the learning. A potential explanation for the absence of details could be attributed to the resolution influenced by the altitude or size of the image. If the resolution has significantly injured the quality of the feature extraction it could be more detrimental to the multi-patch rather than the single patch as there is more patch border and consequently more defective plants could end outside patches which would make them disappear.

The observed disparities in the reconstruction of input data between defective and healthy plants can be attributed to three primary factors. Firstly, the visual quality of the defective plants appears to be inferior when compared to their healthy counterparts. This discrepancy may arise from the way images are exported from the Unity environment, potentially impacted by the correct sizing with the preservation of textures. Secondly, the color of the plant, however, the average Euclidian distance is higher for defective plants which suggests that it is not solely the color. Lastly, the prevalence of healthy plants significantly outweighs that of defective ones, with defective plants accounting for a maximum of 6% of the total plant population. This imbalance in occurrence could influence the model's learning process and subsequently impact its ability to reconstruct defective plant images accurately. This imbalance might be affected by the performance of the on-policy actor-critic, as it is getting better at finding defective plants, the world model inherits from this improvement as the portion of encountered defects should follow the performance. To improve the quality of the masks used in phenotypic error analysis, which show holes, using Morphological operators like open and close filters is advised. The instability seen in the average phenotypic error, particularly with single patches, could be due to the lack of defective plants in the sample, affecting error calculation.

The world model has displayed a strong ability to learn the dynamics of the environment enlightened by the closed-loop results. A constant improvement was visible during the initial steps of the evaluation sequence of the world model, as the agent inherited from the recurrency, coherent with the video prediction objective and the initialization of the recurrent model.  While learning the dynamics of the environment the model exhibits a discernible predictive pattern, revealing a bias towards consistently forecasting a scenario of thriving crops. This aligns coherently with the on-policy behavior of our agent, constrained to remain in the field due to the reward structure. OL and CL could be tested using the same dataset to ensure a more accurate comparison, it would also be interesting to evaluate specific scenarios such as one with healthy plants, defective plants, solely rotating, back, and forth…

The environmental conditions are notably simple, devoid of elements like weeds, trees, or occlusions, suggesting that both the feature extraction process and the policy may have been significantly streamlined. This simplicity likely contributes to the rapid learning behavior of the feature extraction method visible in the closed loop and the learning of the dynamics visible in the open loop, which relies on highly geometric and predictable features. The open loop prediction has displayed unstable error propagation overstep, contradicting the steady increase expected in a dream.  The mathematical generation of the field might have contributed to the performance, by averaging the plants' detail it can well depict how globally any row looks like. Therefore, the error could be attributed to the variability details in the simulation such as missing plants, and offset of plants, which could explain this symptomatic spiking effect. Moreover, field variables such as the distance between rows are fixed which might contribute to good dreaming performance as it just requires knowing the location of one row to depict the other ones. Additional randomness in field variables might contribute to the improvement in generalization and depiction of key features. The dreamer adeptly grasped the environment's dynamics, but the agent's potential to learn robot dynamics was restricted because its actions were discrete, such as momentum.

The performance of the world model to depict the history map was blurry but the main pattern was present. While dreaming the world model was not able to update the map, even more, the error oversteps steadily increased with either disappearing of the path over the dream or a non-updated map. An improvement of the cover map prediction could be the addition of a sigmoid layer, this would ease the reconstruction to a precise 2D binary map. As this layer does not require training it was performed as a post-processing for evaluation purposes, however the performance did not display a significant difference. It would be a better addition to the model so that the world model can learn from this layer by minimizing the loss. Regarding the second channel of the cover map, the world model's inability to accurately depict the current position may be attributed to the insufficient encoding of this information in its latent representation. The cover map also contains a relative redundancy that may contribute to this limitation. Notably, the task of reconstructing a single value in a specific cell might not be suited for the convolutional structure employed, potentially hindering the model's capacity to capture the intricacies of the current position effectively. Moreover, the total loss of this channel might be insignificant compared to the loss of the other channel which had many more used pixels, splitting the channel might be a reasonable solution.

In the original paper (Hafner et al., 2023) the performance of the world model was similar, loss of details in the reconstruction however decent performance in learning the representation and the dynamics. Three noticeable differences are present in the world model, the first-person view, the longer duration of the training and the step at which the environment was learning which was smaller than the one implemented in our experiment.

## 5.2 Actor critic

The Row-by-Row (RbR) method, while heavily optimized but rigid, stands as a benchmark for assessing Dreamer's capabilities. Being the simplest Cover Path Planning (CPP) approach, and widely used in various domains, it is a pertinent comparison. However, the addition of complexities, such as travelling to corners to start the sequence or in the case of a consequent field, could potentially expose the limitations of RbR. In particular, the Dreamer might outcompete RbR, as nearly achieved in scenario 2.1, highlighting the flexibility and adaptability advantages of the Dreamer. As the RbR has outcompeted the DreamerV3 it could seem like the achievements of the Dreamer are not impressive. However, the DreamerV3 has demonstrated incredible potential for usage in the defective plant search especially in the initial phase of search. The dreamer showed increased flexibility by starting at a random position on the edge of the field and was able to perform active perception when confronted with defective plants.

The single-patch and multi-patch scenarios have exhibited distinct performance underlying different exploration strategies. Two distinct types of exploration strategies could be seen. The first type involves general exploration across the entire field, while the second type focuses on in-patch active perception. The multi-patch scenario proves notably more challenging to execute than the single-patch counterpart, demanding increased field exploration while balancing active perception. For the single patch, this balance is not needed as the two strategies are subsequent. Moreover, the bias of the world model might have an impact on the performance, the multi-patch scenario has more borders and therefore more lonely plants outside a main patch. As demonstrated, the world model tends to lose those plants, therefore the multi-patch scenario might have more unreconstructed defective plants. An evaluation of both single and multi-patch detection using one agent could be done. The model could therefore learn a strong and flexible exploration mechanism, while not having prior knowledge of the number of patches which should resemble more to a real-world exploration.
The cover map was a significant addition to the agent’s as it was now able to find defective plants much quicker and more importantly reduce the overlap of its path in the single patch scenario with a total of 450 steps to detect all plants. This could be seen as an increase in global awareness or force memory.

In the original paper (Hafner et al., 2023) the actor-critic also displays great potential in setting new state-of-the-art performance of the tried tasks and being the only agent to collect diamonds in Minecraft.

## 5.3 Limitation:

Within our framework, the world model and actor-critic are linked through latent representation. However, a notable distinction is the lack of shared learning, potentially impacting actor-critic performance. This design choice may hinder the latent representation from effectively extracting essential features for optimal task execution. This challenge mirrors a common issue in integrating learned forward dynamics models with planning and policy learning algorithms (Nair et al., 2020). The misalignment arises from the difference in objectives: the learned model aims at future state reconstruction, while policies focus on task completion. Learning to model what matters (Nair et al., 2020) involves directing the reconstruction toward task-relevant information, making the model task-aware and emphasizing the modelling of pertinent state space aspects for task alignment. The main addition consists of inputting a desired situation in parallel to the input.

In the scope of this research, the chosen top-down camera orientation is a standard use in agriculture. However, this perspective may have imposed constraints on the quality of active perception. While a top-down view ensures consistent resolution and plant state recognition due to a fixed altitude, it may not fully exploit the potential of active perception, which thrives on a dynamic field of view. To enhance active perception, options include adjusting the camera angle, incorporating it into the drone's action space, adding a secondary camera or even using height in the action space. These modifications could expand the drone's operational range and improve field defect detection.

The current evaluation metric, detection overstep may not grasp Dreamer's proficiency in accomplishing the desired task. The average length of exploration could provide insights into the efficiency and effectiveness of the Dreamer in navigating and completing the task more efficiently than Row by Row.

Even if the method used in the dreamer is highly transferrable to any goal-directed task the training is not reputed to be transferable to other environments and robots (Richard et al., 2022). Richard et al found that splitting the state into an environment and a dynamic state allows the use of a physics-informed model which increases the performance and transfers portion of the learning. However, they are required to have the same state space size.
The drone's operational environment in this research was oversimplified with the absence of collision or obstacle. Introducing dynamic elements such as wind and physical obstacles could significantly impact performance since current latent state models do not account for such variables however most drones already display very stable positions. Moreover, dreamers are renowned to excel in learning the dynamics of the environment however tend to overshoot the dynamics of the agent (Richard et al., 2022). The absence of drone dynamics could have skewed the results in favor of the dreamers' proficiency. Further investigation should be made specifically on learning drone dynamics.

To address the possible issues related to the quality of the grand truth image the size of the image could be increased as well as the depth of the encoder, this should solve the precision of the context input. However, the quality of the reconstruction might lack precision as the image was encoded into a fixed-sized latent state. Increasing the total size of the model should therefore be considered if precise reconstruction is targeted. In the original Dreamer V3 paper (Hafner et al., 2023) the author mentioned that the performance and data efficiency increased with the increasing size of the model. Currently, the model is used in its small size configuration with 18B parameters however the original paper has tested bigger models for the more complex task such as in Minecraft. This could raise a source of improvement. The increase in size is notably the number of convolution filters, number of MLP layers, size of latent state… This improvement raises the question of the general-purpose nature of the dreamer as it still requires optimizing the model for the task.

Finally, while DreamerV3's focus on searching for defective plants is beneficial, its failure to cover the entire field could limit its utility for comprehensive crop monitoring.

## 5.4 Future integrations

To advance the use of drones for identifying defects in crop fields, several developments are necessary. Initially, the transition from ground truth-based to an optimized detection algorithm is critical for performing the task and triggering accurate rewards. Moreover, the effectiveness of the drone in detecting subtle early warning signs of crop diseases, which are less visually distinctive than the black markers used in this study, remains to be assessed. An improvement could come from the detection algorithm using segmentation and providing the masks. Furthermore, unlike the simulated environment which resets after identifying all defects, real-world applications lack this convenience. It is essential to evaluate DreamerV3's efficiency in real settings, where it must autonomously determine the cessation of exploration without resets. In a practical application, the drone would fly multiple times in the year/month over the field, a history could therefore be useful for the agent to increase its efficiency over multiple iterations.

Deployments of such agents might raise several issues arising from its highly black box nature, the main one being safety which would require overarching constraints. This approach is the main one for real-world deployment of RL agents which is opposed to safe-RL methods, training the agent to learn to fulfil those safety measures. Dreamers have already been used in the safe RL field and displayed good performance such as Safe Dreamer (Huang et al., 2023).

Regarding future work in agri-food where Dreamer V3 can have a significant contribution as an autonomous agent, it is worth mentioning that solutions applied to real-life scenarios remain rare. However, two main applications stem, active perception and the vision-based decision-making. The first one revolves around the active increase of awareness of a state like our case where we want to inspect a finite space. A major application would be the next best-view algorithm. DreamerV3 offers a solid framework to complete the task with an easily implementable reward of optimizing the discovered voxels.

The second one involves a more stable scenario where the robot must perform a task based on vision-based cues. In the agriculture & food field, those could pick and place robots to manipulate chicken breasts, harvest fruits, sort vegetables, weeding, etc. It can also apply to mobile robots where the goal is not to increase its awareness such as crop driving vehicles from lidar or camera.

On the other hand, I would recommend using Dreamer V3 for short-horizon planning, dreamer has demonstrated exceptional performance in abstracting and predicting its impact on the world for multiple steps. However, it also demonstrated weak points to maintain global awareness for a long time, demonstrated by the highly overlapping path chosen and looping on its path. Further investigation should be taken as it contradicts the original paper (Hafner et al., 2023) claims.

A potential application arises from the ability to predict videos using the world model. Most machine vision and robotic tasks use a single frame to phenotype the environment. However, many applications could inherit the ability to make use of sequences of images inheriting context propagation through time. Tracking algorithms seem a good application to start with, using a first image and initial localization of the target object, the agent should transition to another view and depict the environment and the new position of the target object in the image.

Despite the challenges presented, dreamerV3 displays great potential for the agri-food industry, it showed impressive flexibility regarding the input state and developed strong strategies, especially with single objectives. DreamerV3 offers a solid framework to learn control using multi-sensory inputs.

# 6 Conclusion:

This study focused on evaluating the performance of a latent dynamics model base RL agent (DreamerV3) to search for defective plants in a simulated environment. From this main goal, two experiments were conducted, the first one involved assessing the performance of the Dreamer V3 to search for defective plants with varied defective patch distribution. The second experiment focused on the evaluation of the performance of DreamerV3 with increased context and varying defective patch distribution. Each experiment uses the DreamerV3 architecture to address the performance by first evaluating the performance of the world model to extract features and depict the dynamics, then, the performance of the actor-critic to perform the desired task.

## 6.1 Number of patches

Regarding the first research question, the tested agent displays decent but suboptimal performance in both the world model and the actor-critic. The World model has displayed a lack of precision in reproducing the smaller details where the actor-critic showed a highly overlapping path and was not able to consistently complete the task.

The world model has globally demonstrated reliable performance in abstracting the environment. The row orientations were well encoded while losing details of the plants displayed by averaging the colour, the shadows were well-positioned. The model learned the dynamics of the environments with its actions displaying correctly the shadows, and row orientation. The model updated the field with thriving crops where the mathematical generation of the crop has shown to be well learned by the world model. A lower reconstruction error was seen in the healthy with an average pixel error almost null compared to defective plants with an average pixel error of 0.1, further identification of the cause should be initiated. Moreover, the presence of defective plants outside the main patch tends not to be encoded which suggests the need for refinement of the learning, increase the resolution or addition of variability of the inputs to extract key features of interest.

Even if the model is well able to extract features to represent the world the performance of the dreamer did not outcompete the baseline comparison method. The strategy learned by the actor-critic was sub-optimal and did not find all defects in the allocated time with highly overlapping paths demonstrating the lack of global exploration. The detection rate ended around 0.8 at the end of the runs for both distributions, in double the time of the RbR. The agents were able to perform active perception, however, the multi-patch struggled to balance exploration and active perception correctly.

## 6.2 Increased context

The dreamerV3 upgraded with the cover map has shown to be a powerful agent for defective plant search, outcompeting the baseline comparison method for most (9/10) of the trials. The encoding and learned dynamics of the image have shown a similar pattern to the first sub-research questions. Similarly, for the cover map, the dreamer was able to roughly encode the cover map. The world model was not able to integrate its movement into the global map in the dreaming phase. The current position was not well reconstructed which might result from the incompatibility with the model architecture.

Even if the world model was not able to precisely encode and update the cover map and the image. The additional global awareness has shown to support the exploration and active perception directed to search for defective plants when grouped in a single patch. Efficient strategies were developed to find the patch starting from any position in the corner of the field, nearly outcompeting the RbR with around 450 steps. The multi-patch strategy has shown to be more challenging and did not display improvement in performance ending up with a detection rate of 0.7 in 800 steps.

DreamerV3 has shown to be a powerful agent in searching for defective plants when provided with images and the cover map for a single patch scenario. Other scenarios have shown to be more challenging to learn. While the task was oversimplification the dreamerV3 displays promising opportunities for the agri-food sector. Several improvements have been discussed to support the development of DreamerV3 in defective plant search and the agri-food sector.

# 7 Recommendation:

In our experiments, the action space was discretized, however, when controlling a drone into an efficient path generator to be deployed in agriculture the action space should be modified to control all the degrees of freedom of the drone. Drones don't have any orientation and could make use of it to optimize their path. Currently, the Pitch (forward /backward) and Yaw are used in the action space. Two additional degrees of freedom are used in drone control. The first is the roll tilting to move right and left similarly to the pitch. Finally, height could heavily speed up the performance of the task. Specific attention should however be paid to the detection mechanism as the field of view increases with height but reduces resolution. However, the detection is solely dependent on the position of the plant and visible area. Adding a height threshold to the detection mechanism could be a reasonable solution that might require optimization. A feasible alternative and interesting addition would be to introduce the detection mechanism as an additional action of the agent. This could be done by adding a form of spraying action. The main advantage of this action is that we can now evaluate the performance of the DreamerV3 to detect the plants with a performance matrix. An important addition could be the implementation of the action space as continuous; Learning to control the drone, using lower-level action would complexify the task however it would increase the resemblance to real-world drone control and allow for more simultaneous movement. It could also improve the quality of the realistic dynamics of the drone while including a form of momentum in the drones' dynamics, which must be mastered.

This research has demonstrated that the agent was able to perform the task directed with the reward, however, the policy has displayed some suboptimality displayed by the overlapping of the path in some scenarios. To deal with this we could add a reward to explicitly minimize the overlap. Diverse ways are possible to differentiate by the frequencies of reward, using the end overlap of the path with the average coverage per step or the discovered area at each step or average coverage per step. This strategy would complete the loss traditionally used in coverage path planning explicitly emphasizing the need for exploration with minimal coverage. The reward coefficient could also have necessitated a finer optimization when they are balanced with other rewards or stopping criteria, or even different between actions. Moreover, Dreamer V3 rescales the reward however those are all summed therefore some might still lose impact a multi-head reward prediction might be interesting to investigate.

To achieve high-level performance, we must carefully consider the balance between a model's flexibility and its precision. Despite the claim of the creator of Dreamer V3 to create a general-purpose agent, the performance of the model could be improved by changing hyperparameters such as the learning rate of the world model and the actor. However, it is worth noting that the performance is not linearly related to the number of steps as the agent is trained by acting on the environment using the learned policy. The agent can sometimes be stuck for multiple steps in a suboptimal policy until getting out. Moreover, the actor-critic will have to adjust for the training of the world model which might be computationally intensive if the learning rate is reduced early. Adjustments in training strategies may also be considered to address the observed discrepancy in the reconstruction errors.

The Dreamer displayed significant performance with high flexibility regarding its inputs, which is one main advantage of this world model. The image has shown itself to be powerful, however, the image and the additional cover map have displayed even better performance.  DreamerV3 seems to be well able to integrate multiple senses using images as vision, and the cover map as a form of self-localization and memory. Regarding the current position we could on one hand input it as a separate encoder. On the other hand, we could try to input them as the cartesian coordinates using a multilinear perceptron (MLP), the relationship between the built 2D map and these coordinates will however have to be learned. Other senses could be added such as proprioception using an accelerometer, x/y coordinates, etc. However, it has been clear that the world model is a feature & dynamics extraction method. The question remains for more complex scenes where the precision of the scene might be more important.

Acknowledging the constraint of environment speed, transitioning to an optimized and faster environment would significantly contribute to the evaluation of performance. Optimizing the environment for speed can alleviate potential bottlenecks, providing a more efficient platform for evaluating the agent's capabilities.

The agent has been compared to a hardcoded path planning algorithm; despite the highly optimized nature of this strategy, it might be more accurate to compare it also with other RL algorithms. The agents to consider are the ones supporting a partially observable environment, and images as input or model free algorithm with humanly extracted features.

Recent advancements have changed the way robots are developed where specific tasks are learned, GRID (Vemprala et al., 2023) is a platform for general robot intelligence. It uses large languages model (LLMs) as the orchestrator to incorporate different inputs, leverage the used different model (segmentation, tracking(etc)) to safely control and guide the robot for any drone task. The higher-level focus of the method could offer increased flexibility potentiating the "general purpose" while whitening the model.

# 8 Appendix

## 8.1 Appendix 1: Path of the Row-by-Row method

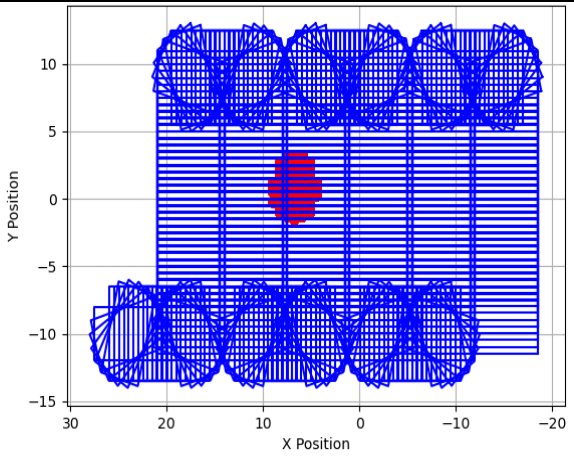
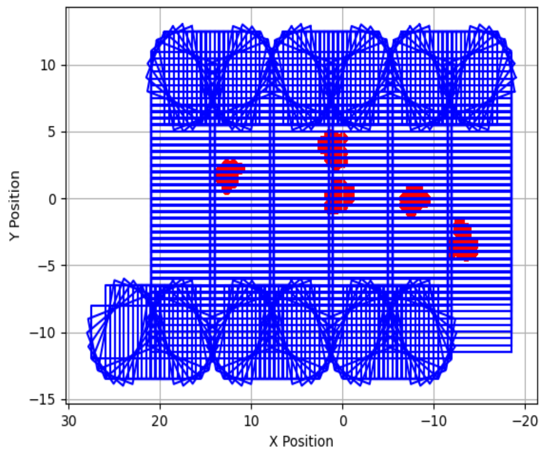
| A | B |
|---|---|
|  |  |

*Figure 36 Row by Row path A: single patch B: multi-patch*

## 8.2 Appendix 2: Multi-patch Phenotypic Evaluation Scenario 1

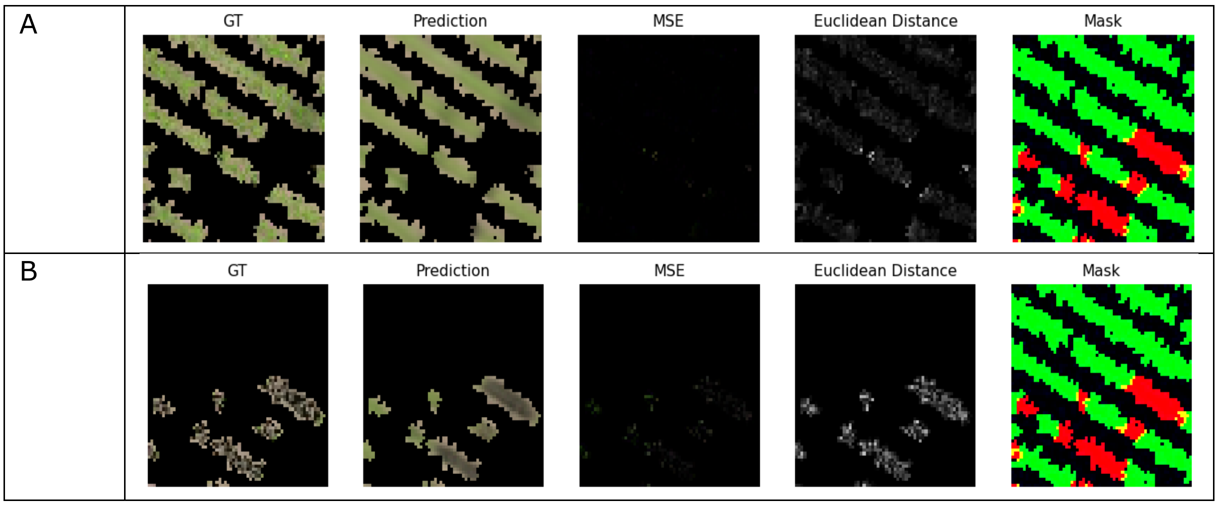


*Figure 37 Phenotypic error scenario 1.2 (A: healthy plants, B: defective plants, the original image (GT) and the predicted image (Prediction) are compared in the MSE and Euclidian distance and finally the mask used is visible (Mask)*

## 8.3 Appendix 3: Multi-patch phenotypic evaluation scenario 2

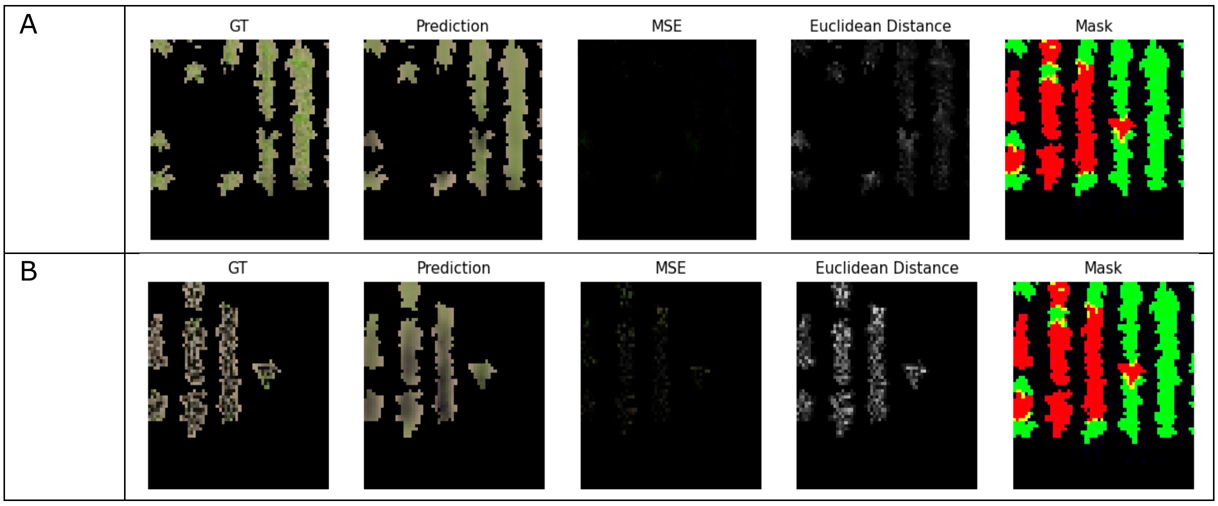


*Figure 38 Phenotypic error scenario 2.2 (A: healthy plants, B: defective plants, the original image (GT) and the predicted image (Prediction) are compared in the MSE and Euclidian distance and finally the mask used is visible (Mask)*

## 8.4 Appendix 4: Normalized error in close loop error cover map

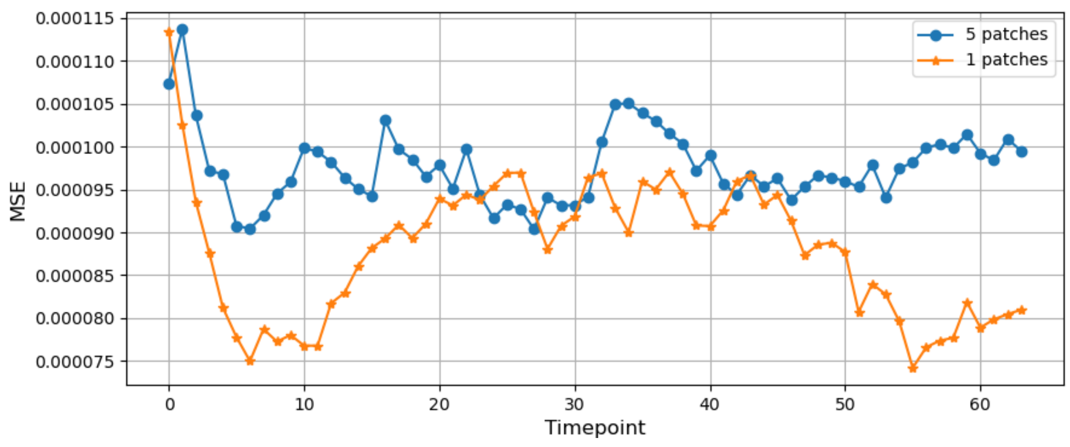


*Figure 39 Normalized error oversteps cover map*

# 9 Bibliography

Ash, G. (2021). *PEST AND DISEASE MANAGEMENT*.

Bai, Y., Zhang, B., Xu, N., Zhou, J., Shi, J., & Diao, Z. (2023). Vision-based navigation and guidance for agricultural autonomous vehicles and robots: A review. In *Computers and Electronics in Agriculture* (Vol. 205). Elsevier B.V. https://doi.org/10.1016/j.compag.2022.107584

Been, T. H., Beniers, J. E., & van der Wolf, J. M. (2022). Determining the Spatial Pattern of Blackleg Diseased Potato Plants in the Field and Simulated Post-Harvest Tuber Infections in Potato Crates. *Potato Research*, *65*(2), 395–416. https://doi.org/10.1007/s11540-021-09529-6

Bocquet-Appel, J. P., Naji, S., Vander Linden, M., & Kozlowski, J. (2012). Understanding the rates of expansion of the farming system in Europe. *Journal of Archaeological Science*, *39*(2), 531–546. https://doi.org/10.1016/j.jas.2011.10.010

Carbone, C., Potena, C., & Nardi, D. (2020). Simulation of near infrared sensor in unity for plant-weed segmentation classification. *SIMULTECH 2020 - Proceedings of the 10th International Conference on Simulation and Modeling Methodologies, Technologies and Applications*, 81–90. https://doi.org/10.5220/0009827900810090

CBS. (2023). *Population counter*. https://www.cbs.nl/en-gb/visualisations/dashboard-population/population-counter

Chadi, M.-A., & Mousannif, H. (2023). *Understanding Reinforcement Learning Algorithms: The Progress from Basic Q-learning to Proximal Policy Optimization*.

Crippa, M., Solazzo, E., Guizzardi, D., Monforti-Ferrario, F., Tubiello, F. N., & Leip, A. (2021). Food systems are responsible for a third of global anthropogenic GHG emissions. *Nature Food*, *2*(3), 198–209. https://doi.org/10.1038/s43016-021-00225-9

Cun, L., Henderson, J., Le Cun, Y., Denker, J. S., Henderson, D., Howard, R. E., Hubbard, W., & Jackel, L. D. (1989). *Handwritten Digit Recognition with a Back-Propagation Network*.

Doster, E., Zitomer, R., & Chislock, M. F. (2013). *Eutrophication: Causes, consequences, and controls in aquatic ecosystems MEGARes: An antimicrobial resistance database for high throughput sequencing View project Enrichment allows identification of diverse, rare elements in metagenomic resistome-virulome sequencing View project*. https://www.researchgate.net/publication/285683019

Ehrlich, A. H. (1995). Implications of Population Pressure on Agriculture and Ecosystems. *Advances in Botanical Research*, *21*(C), 79–80. https://doi.org/10.1016/S0065-2296(08)60009-9

Fevgas, G., Lagkas, T., Argyriou, V., & Sarigiannidis, P. (2022). Coverage Path Planning Methods Focusing on Energy Efficient and Cooperative Strategies for Unmanned Aerial Vehicles. In *Sensors* (Vol. 22, Issue 3). MDPI. https://doi.org/10.3390/s22031235

Gladyshev, M. I., & Gubelit, Y. I. (2019). Green Tides: New Consequences of the Eutrophication of Natural Waters (Invited Review). In *Contemporary Problems of Ecology* (Vol. 12, Issue 2, pp. 109–125). Pleiades Publishing. https://doi.org/10.1134/S1995425519020057

Ha, D., & Schmidhuber, J. (2018). *World Models*. https://doi.org/10.5281/zenodo.1207631

Haarnoja, T., Zhou, A., Abbeel, P., & Levine, S. (2018). *Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor*. http://arxiv.org/abs/1801.01290

Hafeez, A., Husain, M. A., Singh, S. P., Chauhan, A., Khan, M. T., Kumar, N., Chauhan, A., & Soni, S. K. (2023). Implementation of drone technology for farm monitoring & pesticide spraying: A review. In *Information Processing in Agriculture* (Vol. 10, Issue 2, pp. 192–203). China Agricultural University. https://doi.org/10.1016/j.inpa.2022.02.002

Hafner, D., Lillicrap, T., Ba, J., & Norouzi, M. (2019). *Dream to Control: Learning Behaviors by Latent Imagination*. http://arxiv.org/abs/1912.01603

Hafner, D., Lillicrap, T., Fischer, I., Villegas, R., Ha, D., Lee, H., & Davidson, J. (2018). *Learning Latent Dynamics for Planning from Pixels*. http://arxiv.org/abs/1811.04551

Hafner, D., Pasukonis, J., Ba, J., & Lillicrap, T. (2023). *Mastering Diverse Domains through World Models*. http://arxiv.org/abs/2301.04104

Hajkowicz, S., Sanderson, C., Karimi, S., Bratanova, A., & Naughtin, C. (2023). Artificial intelligence adoption in the physical sciences, natural sciences, life sciences, social sciences and the arts and humanities: A bibliometric analysis of research publications from 1960-2021. *Technology in Society*, *74*. https://doi.org/10.1016/j.techsoc.2023.102260

Huang, W., Ji, J., Zhang, B., Xia, C., & Yang, Y. (2023). *SafeDreamer: Safe Reinforcement Learning with World Models*. http://arxiv.org/abs/2307.07176

Isenring, R. (2010). *Pesticides and the loss of biodiversity How intensive pesticide use affects wildlife populations and species diversity.* www.pan-europe.info

Moisa, M. B., Dejene, I. N., Hirko, O., & Gemeda, D. O. (2022). Impact of deforestation on soil erosion in the highland areas of western Ethiopia using geospatial techniques: a case study of the Upper Anger watershed. *Asia-Pacific Journal of Regional Science*, *6*(2), 489–514. https://doi.org/10.1007/s41685-022-00238-7

Mooney Sacha, Cooper Hannah Victoria, & Sjogersten Sofie. (2023). *Farming without disturbing soil could cut agriculture's climate impact by 30% – new research*. https://theconversation.com/farming-without-disturbing-soil-could-cut-agricultures-climate-impact-by-30-new-research-157153

Nair, S., Savarese, S., & Finn, C. (2020). *Goal-Aware Prediction: Learning to Model What Matters*. http://arxiv.org/abs/2007.07170

NM512. (2023). *dreamerv3-torch*. https://github.com/NM512/dreamerv3-torch?tab=MIT-1-ov-file#readme

Rehder, L. (2022). *EU Consumers save on food and buy less organic in 2022_Berlin_European Union_E42023-0005*.

Rejeb, A., Abdollahi, A., Rejeb, K., & Treiblmaier, H. (2022). Drones in agriculture: A review and bibliometric analysis. In *Computers and Electronics in Agriculture* (Vol. 198). Elsevier B.V. https://doi.org/10.1016/j.compag.2022.107017

Richard, A., Aravecchia, S., Geist, M., & Pradalier, C. (2022). *Learning Behaviors through Physics-driven Latent Imagination*.

Ryan, M., Jansen, M., Nielsen, J., Gruère, G., Antón, J., Asai, M., Directorate, A., Quintini, G., & Goglio, A. (2023). *LABOUR AND SKILLS SHORTAGES IN THE AGRO-FOOD SECTOR OECD TRADE AND AGRICULTURE DIRECTORATE Labour and Skills Shortages in the Agro-Food Sector*.

Said Mohamed, E., Belal, A. A., Kotb Abd-Elmabod, S., El-Shirbeny, M. A., Gad, A., & Zahran, M. B. (2021). Smart farming for improving agricultural management. In *Egyptian Journal of Remote Sensing and Space Science* (Vol. 24, Issue 3, pp. 971–981). Elsevier B.V. https://doi.org/10.1016/j.ejrs.2021.08.007

Sharma, M., & Hema, N. (2021). Comparison of Agricultural Drones and Challenges in Implementation: A Review. *2021 7th International Conference on Signal Processing and Communication, ICSC 2021*, 26–30. https://doi.org/10.1109/ICSC53193.2021.9673491

Steenwyk, P., Heun, M. K., Brockway, P., Sousa, T., & Henriques, S. (2022). The Contributions of Muscle and Machine Work to Land and Labor Productivity in World Agriculture Since 1800. *Biophysical Economics and Sustainability*, *7*(2). https://doi.org/10.1007/s41247-022-00096-z

Sutton, R. S., & Barto, A. G. (1998). Reinforcement Learning: An Introduction. *IEEE Transactions on Neural Networks*, *9*(5), 1054. https://doi.org/10.1109/TNN.1998.712192

Sutton, R. S., & Barto, A. G. (2018). *Reinforcement Learning An Introduction second edition*.

Sutton, R. S., Mcallester, D., Singh, S., & Mansour, Y. (1999). *Policy Gradient Methods for Reinforcement Learning with Function Approximation*.

Sylvester, G. (2018). *E-agriculture in Action: Drones for Agriculture*.

Tolman, E. C. (1948). *THE PSYCHOLOGICAL REVIEW COGNITIVE MAPS IN RATS AND MEN ** (Vol. 55, Issue 4).

Touvron, H., Lavril, T., Izacard, G., Martinet, X., Lachaux, M.-A., Lacroix, T., Rozière, B., Goyal, N., Hambro, E., Azhar, F., Rodriguez, A., Joulin, A., Grave, E., & Lample, G. (2023). *LLaMA: Open and Efficient Foundation Language Models*. http://arxiv.org/abs/2302.13971

Vacante, V., & Bonsignore, C. P. (2012). Implementation of IPM in citriculture. In *Integrated Control of Citrus Pests in the Mediterranean Region* (pp. 28–55). Bentham Science Publishers Ltd. https://doi.org/10.2174/9781608052943112010100 28

Van Henten, E. J., Hemming, J., Van Tuijl, B. A. J., Kornet, J. G., Meuleman, J., Bontsema, J., & Van Os, E. A. (2002). An Autonomous Robot for Harvesting Cucumbers in Greenhouses. In *Autonomous Robots* (Vol. 13).

Vemprala, S., Chen, S., Shukla, A., Narayanan, D., & Kapoor, A. (2023). *GRID: A Platform for General Robot Intelligence Development*. http://arxiv.org/abs/2310.00887

Watkins, C. J. C. H., & Dayan, P. (1992). *Q-Learning* (Vol. 8).

Zhang, W., Yao, Y., Xiao, X., Balasubramanian, V. N., & Tsang, I. (2021). Robust Model-based Reinforcement Learning for Autonomous Greenhouse Control. In *Proceedings of Machine Learning Research* (Vol. 157).

Zhao, T., Wang, Y., Sun, W., Chen, Y., Niub, G., & Sugiyama, M. (2022). *Representation Learning for Continuous Action Spaces is Beneficial for Efficient Policy Learning*. http://arxiv.org/abs/2211.13257

Cover page image:
Wonderfull.ca, 2024, Farm Tario, https://farmtario.com/machinery/can-drones-replace-self-propelled-spraying-equipment/