# Model-Based Novelty Adaptation for Open-World AI

**Matthew Klenk**[1] and **Wiktor Piotrowski**[1] and **Roni Stern**[1,2] and **Shiwali Mohan**[1] and **Johan de Kleer**[1]
[1]Palo Alto Research Center, CA, USA
e-mail: {matthew.klenk,wiktorpi,rstern,mohan,dekleer}@parc.com
[2]Ben-Gurion University of the Negev, Beer-Sheva, Israel

## Abstract

To be able to function in an open-world, a system must be able to adapt to novel situations. We consider this problem in the context of the popular Angry Birds game and outline our system design to address it. Our system, called *Hypothesis-Guided Model Revision over Multiple Aligned Representations* (HYDRA), adopts a model-based approach to respond to novelty. Central to our design is the use of mixed continuous-discrete planning formalism, namely PDDL+ [Fox and Long, 2006], to model the Science Birds domain. With this model, HYDRA employs a domain-independent planner to play the game, and model-based diagnosis to diagnose and repair the model when novelties are introduced. We also report a case study demonstrating how HYDRA adapts its domain theory to changing dynamics in ballistic flight.

## 1 Introduction

One hallmark of human cognition is our ability to function in an open-world. People navigate to previously unseen places, perform new tasks, and integrate new technology into their lives. In games, human flexibility supports inventing new strategies along with adapting to changing rules (e.g., consider chess players who play bughouse[1]). While current AI systems perform superhuman in many game domains, each of these domains is a closed world, and minor perturbations of the game can lead to significant drops in performance. Witty et al. demonstrated that even changes which made the game easier could cause catastrophic results for superhuman performing deep Q-learning agents [Witty *et al.*, 2018]. This mismatch between human cognitive abilities and machine capabilities indicates that adapting to novelty is a major problem in current AI systems. Our research goal is to impart machines with human-like learning capabilities to make them robust to novelty.

In this paper, we present the novelty response problem problem in the context of Science Birds domain. Then, we introduce *Hypothesis-Guided Model Revision over Multiple Aligned Representations* (HYDRA), our approach to model-based novelty response. We provide an outline to our system design, which consists of a model-based diagnosis component to detect, diagnose, and repair its underlying model when predictions from models differ from observations in the environment.

Central to our design is the use of mixed continuous-discrete planning formalism, namely PDDL+ [Fox and Long, 2006], to model the Science Birds domain. We demonstrate how this enables HYDRA to play the game as well as adapt to many types of novelty by making localized modifications to the domain theory. Next, we present a case study demonstrating how HYDRA adapts its domain theory to changing dynamics in ballistic flight. We close with a discussion of different issues we expect to address in the course of this project.

## 2 Problem Definition

Science Birds is a free version of the popular Angry Birds game. The player launches birds in sequence at a structure made out of different material blocks with the goal of destroying the pigs inside. Different birds and structures have different actions (e.g., yellow birds accelerate when the player taps the screen during flight) and properties (e.g., TNT objects explode when damaged by birds or other falling objects). Science Birds is a challenging domain for AI agents due to the continuous action space and the large state space of resulting block configurations, and has maintained a yearly competition since 2012 [Renz *et al.*, 2019].

Our agent interacts with the game through a server with the following API. After the level is loaded, the agent is given a list of objects with their outer hull polygons and a color-map that specifies the amount of each color in inside the polygon[2]. The agent specifies shots by providing an $(X, Y)$ position to launch from and a time $t$ to tap the screen. The screen tap initiates actions based on bird type (e.g., a bomb bird will explode a few seconds after it is tapped). After each action, the score is updated.

We are studying novelty as something that is introduced into the environment while an agent is performing tasks. In the context of Science Birds, the agent plays a sequence of levels. At some point in the sequence, novelty is introduced and all subsequent levels behave with the novelty. An example of novelty is the introduction of a new bird type with different dynamics and actions that would be available in future levels. Our objective is to play the game, *detect* the novelty when it occurs, and *respond* to it. The result of this

[1]`https://en.wikipedia.org/wiki/Bughouse_chess`

[2]Raw pixels for the entire image are also available, but we do not use them in our system.


Figure 1: A sample Science Birds problem (left) and relevant metrics (right). While we expect that a SOTA agent will outperform novelty responsive AI systems (e.g., HYDRA) as it is would be tailored to the particular domain, we expect HYDRA to recover more quickly after novelty is introduced.

learning will enable our agent to mitigate the effects of the novelty on its performance and, when possible, take advantage of new opportunities available due to the change in the environment on future problems in the sequence. Figure 1 illustrates this process and how we intend to measure performance against a state-of-the-art AI system that is not designed to respond to novelty.

## 3 Proposed Approach

Figure 2 shows an overview of our proposed approach. Science Birds provides the score for the level and a description of the objects. HYDRA classifies these objects into types in its domain theory, and assesses if they have behaved consistently with the domain theories expectations. These expectations could be driven by quantitative or qualitative composable models. Any inconsistencies are localized to model components using model-based diagnosis and learning problems are formulated. For example, if HYDRA does not understand why a structure has not fallen over, a possible explanation is that there is an unseen rigid object supporting it. Then, HYDRA may generate a plan to satisfy the learning goal by shooting a bird in that area and look for evidence of rigid object mechanics.

## 4 Adapting to Novelty with PDDL+

HYDRA's approach is centered around a *planning* module tasked with solving Science Birds levels. Science Birds is an interesting planning problem containing non-linear dynamics as well as both discrete and continuous behaviour. Unlike many other planning problems where most world changes are the direct result of agent actions, Science Birds dynamics are governed by chains of reactions triggered by agent actions. These reactions are difficult to predict without modeling the physics of the Science Birds world.

Due to these properties, we chose PDDL+ [Fox and Long, 2006] as the planning formalism for HYDRA. PDDL+ allows modelling of the environment, its dynamics and behaviour, as well as the agent's interactions with the environment. The defining characteristic of PDDL+ is the ability to model exogenous behaviour with discrete events and continuous processes. Events apply discrete effects instantaneously, whereas processes apply changes continuously while their preconditions hold. The agent has no direct control over processes and events, and can only interact with exogenous activity indirectly. As noted above, Science Birds is overwhelmingly governed by processes and events. Thus, PDDL+ is an attractive language for the Science Birds domain theory.

To date, we have created a PDDL+ model that solves a variety of Science Birds levels. However, planning in PDDL+ can result in search-space explosion due to the tight integration of planning and scheduling over a continuous timeline. To improve the performance, our Science Birds model relies heavily on the *Theory of Waiting* [McDermott, 2003] and currently employs only one action responsible for the release of the bird from the slingshot. This reduces the number of decision points in the search, which significantly reduces the branching factor. For the dynamics, events represent collisions between birds, pigs, blocks, platforms, TNT blocks, and the ground, whereas processes capture the ballistic motion of birds under gravity and changing the possible angle of launch. When our agent receives a Science Birds level to play, it automatically translates it to a PDDL+ planning problem under our Science Birds PDDL+ model. Then, we use an off-the-shelf PDDL+ planner, UPMurphi [Della Penna *et al.*, 2009], to obtain a plan.

### 4.1 Hypothesis-Guided Model Revision

An advantage of having a PDDL+ model is that it enables simulating the expected state of the world over time after an action is performed. HYDRA leverages this capability to detect novelty, as follows. After HYDRA performs an action, it observes the game and collects periodic states from the game API. Then, HYDRA checks if this sequence of states is consistent with the sequence of states it expected to observe according to the model. A novelty is detected when the discrepancy between the observed and expected sequence of states exceeds a predefined threshold.

Following Langley's recent *Theory of Environmental Change* [Langley, 2020], we view novelty as a *transformation* of the underlying world model. To adapt to novelty, HYDRA must update its domain model. To accomplish this, it searches for a hypothesis about the transformations that would be consistent with the observations. HYDRA uses a set of *Model Manipulation Operators* to transform the PDDL+ domain theory. To check if a sequence of MMOs is consistent with the observations, we apply them to the current PDDL+ model, simulate the expected sequence of states according the modified model, and check if this sequence of states is consistent with the sequence of states

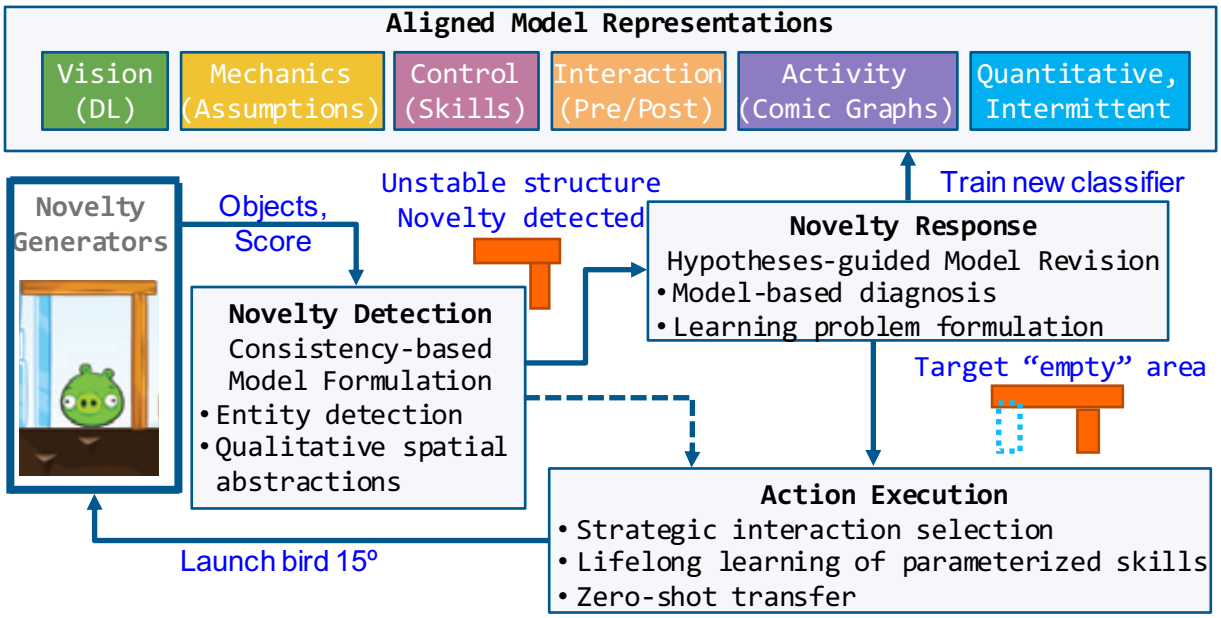


Figure 2: The HYDRA architecture draws on multiple model representations to plan actions, observe their effects, and focus learning.

| Novelty types | PDDL+ domain adjustment | Novelty example in Science Birds |
|---|---|---|
| Spatio-temporal Transformation | Fluent changes | Increased the force of gravity |
| Structures Transformation | New objects and fluents | Introduced new type of bird |
| Processes Transformation | New and/or changing existing processes | Introduced wind |
| Constraints Transformation | New preconditions and/or changed events | Only explosions can kill pigs |

Table 1: Description of example novelties that can be encountered in Science Birds, changes to the PDDL+ model required to accommodate them, and their corresponding novelty types defined by [Langley, 2020].

observed in the game. After a consistent model has been found, it is used by HYDRA to generate future plans.

There may be multiple models consistent with the current observations. Also, new novelties may occur over time. Therefore, the process of detecting novelties and adapting HYDRA's PDDL+ model to them is continuous: after every action HYDRA performs, it checks if the current observation is consistent with its model. If it is not, it searches for a sequence of MMOs that would yield a model that is consistent with the current and previously collected observations.

### 4.2 Searching for Consistent PDDL+ Models and Applicable MMOs

A future objective of this work is to characterize the necessary and sufficient types of MMOs that are needed to adapt to different types of novelties. In our current implementation, we focused in simple MMOs that modify the value of constant fluents in the PDDL+ model such as the force gravity applies on flying objects, the size of the birds, and the speed in which the slingshot's angle is adjusted. Table 1 maps possible types of MMOs to types of novelties as defined by Langley [Langley, 2020] along with examples from Science Birds.

The number of MMOs may be very large and thus finding a sequence of MMOs that may yield a consistent model is a challenging combinatorial search problem. We expect to need heuristics to guide the search in an efficient manner. In our current implementation, we run a Greedy Best-First Search algorithm that uses a heuristic that prefers shorter sequences of MMOs that yield models that are more consistent.

### 4.3 Case Study: Auto-Tuning Gravity

To demonstrate how HYDRA works, we performed the following case study. The agent is given a simple Science Birds level shown in Figure 3, in which it needs to hit a pig that is elevated on some platform. We intentionally set the agent's PDDL+ model to be incorrect by setting the force it assumes gravity applies on objects to be significantly higher than its real value. Using this incorrect PDDL+ model, the agent fails to create a plan that hits the pig, since it cannot throw the bird strong enough to overcome the force of gravity it assumes. In such a case, the agent chooses an arbitrary action, which in this case was to throw the bird at a very high angle. The resulting trajectory is shown in Figure 3 (left). Then, HYDRA uses the observed trajectory of the bird to correct its PDDL+ model. Specifically, the MMOs we used were to modify the gravity parameter by either adding or subtracting 30 from its value. HYDRA uses these MMOs to search for a PDDL+ model that is consistent with the observed trajectory. In this case, HYDRA is able to find such a model, modifying its gravity parameter to a value that is much closer to the correct value. Using the revised model, HYDRA is now able to create a plan that accurately shoots the pig and wins the game, as shown in Figure 3 (right).

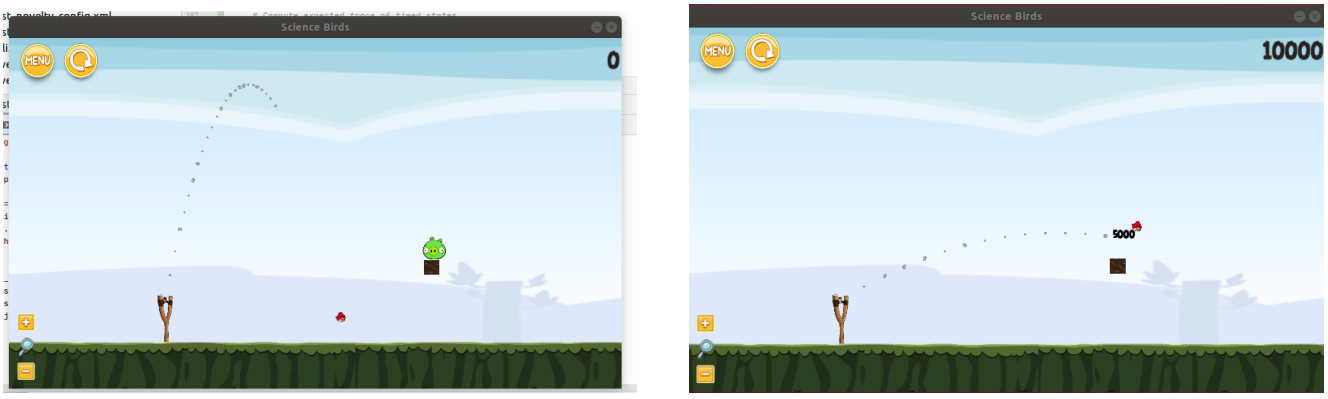


Figure 3: Example of automated model-repair with HYDRA. The left figure shows the first shot, in which HYDRA assumes an incorrect gravity factor. The right figure shows the second shot, after HYDRA diagnosed its incorrect assumption about gravity and corrected it accrodingly.

## 5 Discussion

This early stage work opens up a number of research questions:

1. Are MMOs and search heuristics domain independent? That is, as we transition the technique to other domains (e.g., Minecraft, inverted pendulum control, and simulated driving) will the MMO's change?
2. How much of the domain revisions will be done within the PDDL+ model versus in other models in the system? For example, while the classification task of mapping observations to types is not performed in PDDL+, the types themselves are.
3. How to incorporate agent experience in the model revision decisions? Since our model is an approximation of the world, constantly revising it due to noise would not make sense.
4. How to account for other agents? We propose to modeling the behavior of other agents through their changing configurations with other objects in the environment, a model representation we call *comic graphs* [Klenk *et al.*, 2017].
5. How to integrate PDDL+ planning with reinforcement learning techniques? Parameterized skills [Rostami *et al.*, 2020] provide a method for learning detailed action models that may be organized using planning.

As part of the DARPA SAIL-ON effort, we will explore these questions over the next three years.

## Acknowledgments

This work was supported by the DARPA SAIL-ON program under contract HR001120C0040. The views and conclusions in this document are those of the authors and should not be interpreted as representing the official policies, either expressly or implied, of the Defense Advanced Research Projects Agency or the U.S. Government

## References

[Della Penna *et al.*, 2009] Giuseppe Della Penna, Daniele Magazzeni, Fabio Mercorio, and Benedetto Intrigila. Upmurphi: a tool for universal planning on pddl+ problems. In *Nineteenth International Conference on Automated Planning and Scheduling*, 2009.

[Fox and Long, 2006] Maria Fox and Derek Long. Modelling mixed discrete-continuous domains for planning. *Journal of Artificial Intelligence Research*, 27:235–297, 2006.

[Klenk *et al.*, 2017] Matthew Evans Klenk, Shiwali Mohan, Johan De Kleer, Daniel G Bobrow, Tom Hinrichs, and Ken Forbus. Collaborative autonomy through analogical comic graphs. In *Workshops at the Thirty-First AAAI Conference on Artificial Intelligence*, 2017.

[Langley, 2020] Pat Langley. Open-World Learning for Radically Autonomous Agents. *AAAI*, 2020.

[McDermott, 2003] Drew V McDermott. Reasoning about autonomous processes in an estimated-regression planner. 2003.

[Renz *et al.*, 2019] Jochen Renz, XiaoYu Ge, Matthew Stephenson, and Peng Zhang. Ai meets angry birds. *Nature Machine Intelligence*, 1(7):328–328, 2019.

[Rostami *et al.*, 2020] Mohammad Rostami, David Isele, and Eric Eaton. Using task descriptions in lifelong machine learning for improved performance and zero-shot transfer. *Journal of Artificial Intelligence Research*, 67:673–704, 2020.

[Witty *et al.*, 2018] Sam Witty, Jun Ki Lee, Emma Tosch, Akanksha Atrey, Michael Littman, and David Jensen. Measuring and characterizing generalization in deep reinforcement learning. *arXiv preprint arXiv:1812.02868*, 2018.