The Difficulty of Novelty Detection in Open-World Physical Domains: An Application to Angry Birds

Vimukthini Pinto[1], **Cheng Xue**[1], **Chathura Gamage**[1], **Matthew Stephenson**[2], **Jochen Renz**[1]

[1]School of Computing, The Australian National University, Canberra, Australia. [2]College of Science and Engineering, Flinders University, Adelaide, Australia.
vimukthini.inguruwattage@anu.edu.au

Abstract

Detecting and responding to novel situations in open-world environments is a key capability of human cognition and is a persistent problem for AI systems. In an open-world, novelties can appear in many different forms and may be easy or hard to detect. Therefore, to accurately evaluate the novelty detection capability of AI systems, it is necessary to investigate how difficult it may be to detect different types of novelty. In this paper, we propose a qualitative physics-based reasoning method to quantify the difficulty of novelty detection focusing on open-world physical domains. We apply our method in the popular physics simulation game (PSG) Angry Birds and conduct a user study across different novelties to validate our method. Results indicate that our calculated detection difficulties are in line with those of human users.

1 Introduction

With the increasing reliance on autonomous systems such as self-driving cars and planetary robots, detection and adaptation to novel situations have become important capabilities for such AI systems. For example, if an autonomous car is not trained on slippery roads, the car may fail to detect when the friction is reduced and adjust the speed accordingly. Open-world learning is an emerging research area that attempts to address the challenge of detecting and adapting to novel situations (Langley 2020). Open-world learning research requires adequate evaluation protocols to capture the performance of agents under the two tasks: detection and adaptation (Senator 2019). This paper focuses on creating a difficulty measure for novelty detection to aid the evaluation of novelty detection by disentangling agents' performance from the intrinsic difficulty of novelties.

The novelties we encounter in an open world can take various forms (Boult et al. 2021). In this paper, we focus on *structural transformation*, a very common type of real-world novelty where an unknown object is encountered or a previously known object changes one/more of its properties (Langley 2020). For example, this could be a new vehicle type on the road, a new type of product in the supermarket with new packaging, a previously empty box filled with goods, or an abnormally heavy ball in a billiards game. As these examples suggest, only some of the novelties can be identified from appearance. Novel objects with different appearances can be detected by observing the change in color or shape. Quantifying the difficulty of detecting them can be addressed with the use of concepts presented in color science (Giesel and Gegenfurtner 2010) and research conducted on object shapes and sizes (Perner 2018). However, the difficulty of detecting novel objects with the same appearance but different physical parameters (e.g., mass, friction, bounciness) is not addressed so far. It is also not straightforward as one needs to interact with the objects and observe changes in their movements. Moreover, the detectability of such novelty depends on several factors: the physical parameter that is changed or the number and arrangement of novel objects in the environment.

This paper presents a qualitative-physics based method to quantify the difficulty of detecting novel objects with the same appearance but altered physical parameters (compared to previously seen versions of the objects). The proposed method aids a thorough evaluation by disentangling agents' performance from the difficulty of detecting the novelty. For example, if the novelty cannot be identified from the appearance and occurs in an object that is not reachable to interact, then the novelty cannot be detected. Therefore, the difficulty of novelty detection should be considered before making conclusions on the detection ability of an agent. The method we propose is agent-independent and enables us to evaluate an agent's performance (both detection performance and task performance) at different levels of difficulty (that can be categorized as easy, medium, and hard). We apply our method to the popular PSG Angry Birds, as it has semi-realistic physics and provides an ideal platform to introduce novelty (Gamage et al. 2021). We then conduct a user study experiment with human participants to verify that the calculated novelty detection difficulty values are in line with those of humans.

2 Background and Related Work

This section presents the notion of difficulty and novelty in the context of physical worlds and AI. We also discuss the related work in qualitative physical reasoning and a brief description of our experimental domain.

Difficulty Assessing difficulty is a popular research area in neuroscience where researchers are interested in quantifying the difficulty of tasks or decisions (Franco et al. 2018).

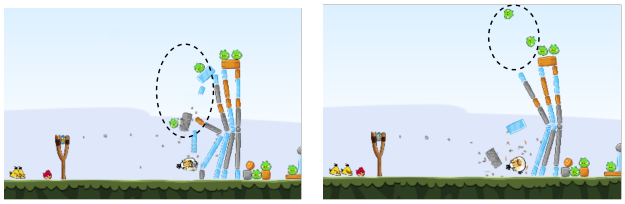

Figure 1: Examples from Angry Birds. The figure in the left (a) has original parameters whereas the figure in the right (b) has an increased *bounciness* parameter for pigs. The two figures show the difference in pig's movement for the same shot in the original (a) and increased *bounciness* (b) of pigs.

Measuring difficulty is also a main topic of discussion when measuring the intelligence of AI systems (Chollet 2019). It is also a widely studied topic in the gaming industry to make games interesting to players. The flow-theory, one of the most prevalent models in the game design, suggests that the games should not be too easy or too difficult to maintain player enthusiasm (Takatalo et al. 2010). Considering the difficulty of detection, researchers have studied this in areas such as emotion detection (Laubert and Parlamis 2019) and missing content detection (Yom-Tov et al. 2005). However, to our best knowledge, the difficulty of novelty detection in physical domains is not studied so far and is important when evaluating the detection capabilities of agents.

Novelty In the context of AI, novelty is described as situations that violate implicit or explicit assumptions about the agents, the environment, or their interactions (SAIL-ON-BBA 2019). Similarly, Boult et al. formalize a theory of novelty for open-world environments and Langley explains different types of environmental transformations that can be considered as novelty. Following these ideas, the novelties we consider in this paper occur as a result of changed physical parameters of objects. It could be the mass, friction, elasticity, etc. These novelties do not change the appearance of the object but cause it to behave differently after an interaction. For example, in the real-world, a novelty could be a new tennis ball with higher bounciness than the balls seen before, a previously empty bottle now filled with water, or a box of goods with less weight due to a manufacturing defect. Figure 1 shows differences in the observed movements of objects after physical parameters have been changed in the research clone of Angry Birds (Ferreira and Toledo 2014).

Qualitative Physics As discussed previously, novelties based on physics parameters are not detectable from appearance alone. Therefore, one needs to interact with the objects and observe any difference in their expected movements. Humans are often unaware of the exact physical parameters such as density, friction, and mass distribution of objects and do not need to solve complex differential equations when reasoning about their movements, instead relying on spatial intelligence (Walega, Zawidzki, and Lechowski 2016).

To analyze object movements, a qualitative physics approach was proposed to approximate structural stability based on the extended rectangular algebra (ERA) (Zhang and Renz 2014). ERA comprises 27 interval relations based on the approximated center of mass of the object and offers more flexibility than the original 13 interval algebra relations (Allen 1983). Ge, Renz, and Zhang point out that ERA is more suitable to approximate the stability of a single object rather than a structure and extends the use of ERA by proposing two qualitative stability analysis algorithms: *local stability* and *global stability*. A similar algorithm, *vertical impact* is proposed by Walega, Zawidzki, and Lechowski, which combines the concepts of *local stability* and *global stability* into one algorithm. They also introduced the algorithm *horizontal impact*, to provide a heuristic value to the interaction based on force propagation. Our difficulty measure also uses the algorithm *vertical impact* along with new reasoning components which reason about the nature of the object movements that are necessary to detect novelty.

Experimental Domain Our experimental domain, Angry Birds is a PSG where the player shoots birds at pigs from a slingshot. These pigs are often protected by different physical structures that are made up of three types of materials: wood, ice, and stone. There are also static platforms, which are indestructible objects that are not affected by forces. The task of an agent that attempts to detect novelty is to identify if anything has changed from the normal game environment by shooting at game objects. As the original Angry Birds game by Rovio Entertainment is not open source, we conduct our experiments using a research clone of the game (Ferreira and Toledo 2014). One example of novelty in Angry Birds is displayed in figure 1. As Figure 1a shows, the agent who is familiar with the normal game environment expects the pigs to fall down without bouncing up after an interaction. However, when the bounciness parameter is increased, the agent observes a change in the pigs' movement as shown in Figure 1b.

We selected Angry Birds as our experimental domain due to three reasons. First, solving an Angry Birds game instance (game level) requires reasoning about object movements in complex physical structures (Zhang and Renz 2014). Second, there are many game levels and level generators (Stephenson et al. 2019) that enable us to evaluate our difficulty measure on a diverse set of levels. Third, this is an ideal platform to vary different physics parameters and add the type of novelties we are investigating in this study. Moreover, Angry Birds is a very popular domain among AI researchers with several long-running competitions associated with it (Renz et al. 2015, 2019).

3 Overview of the Difficulty Measure

This section presents a high-level overview of our difficulty measure formulation for novelty detection in physical domains. We define the following to aid our explanations.

- Each object consists of a set of appearance-related parameters and a set of physical parameters. There is a predefined many-to-one mapping from appearance parameters to physical parameters (objects with the same appearance always have the same physical parameters and two or more objects with different appearance can have the same physical parameters). Objects with the same appearance are referred to as an object type. The number of object types is predefined.

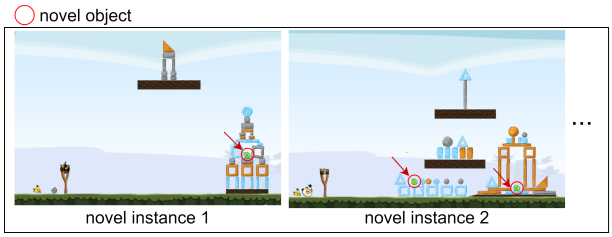

Figure 2: A set of novel instances. Each instance contains one/-more novel pigs denoted by the red circle and a set of normal objects. Note that, this paper focuses on novel objects with the same appearance as non-novel objects but with different physical parameters. Therefore, novel objects cannot be distinguished visually.

- *normal object*: An object that preserves the predefined mapping between appearance and physical parameters.
- *novel object*: An object that violates the predefined mapping between appearance and physical parameters.
- *normal instance*: An arrangement with a collection of *normal objects*.
- *novel instance*: An arrangement with a collection of *normal objects* and *novel objects*. (See Figure 2)

Our measure has three properties. Our difficulty measure:

1. is instance-based, i.e., we provide the difficulty of detecting novelty for a specific novel instance.
2. quantifies the difficulty of detecting novelty when an agent encounters the novel instance for the first time (the agent does not attempt the instance multiple times).
3. is agent independent (i.e., we do not collect data from agents to develop the measure).

Given below are three assumptions we make.

1. As designers of the difficulty measure, we have full understanding of the novel instance (i.e., the novel object, location of the novel object, the changed parameters, and the value of the parameters).
2. The agent has a full understanding of the object dynamics in the normal environment. The agent is fully aware of how objects move without novelty and the agent can detect that the environment is novel if a change in movements happens in the novel environment (perfect detection). We made this assumption because the detection difficulty can be different across agents; therefore, our measure is based on the lower bound of the detection difficulty by assuming perfect detection.
3. The agent attempts to detect novelty using a sequence of interactions. i.e., the agent cannot have multiple interactions at the same time. For example, in the billiards game, an agent can move only one ball at a time.

Figure 3 shows the components of our difficulty measure formulation.There are two inputs, the initial state of an instance (i.e., the state of an instance before any interactions) and the novelty present in the instance. Novelty present can be a set of objects with their changed physical parameter (e.g. {(wood objects, mass), (stone objects, friction)}). Our first module, the *Target Determining Module* takes the above

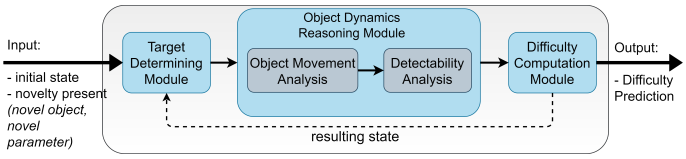

Figure 3: Overview of the method to compute the difficulty of novelty detection.

two inputs and searches possible target objects, i.e., the objects an agent can interact with. This module outputs all possible target objects in the given state.

The second module, *Object Dynamics Reasoning Module* has two components, the *object movement analysis* component and the *detectability analysis* component. The *object movement analysis* component takes each target object and identifies other objects that are moved due to the interaction with the target object. Next, the *detectability analysis* component determines if the interaction has caused the novel object to move in a detectable way. For example, when a novel object has a different sliding friction value, an interaction that causes the novel object to fall from above would not make the novel object detectable. In contrast, an interaction that causes the novel object to slide on a surface would make the novel object detectable.

Knowing the target objects that make detectable movements, the *Difficulty Computation Module* quantifies the difficulty of novelty detection to the given initial state. If the algorithms in the difficulty computation module require the next state to predict the difficulty, the updated states (i.e., state after an interaction) are sent to the *Target Determining Module* (dotted arrow in Figure 3) and the process iterates until the difficulty for the instance can be calculated.

4 Difficulty Measure Applied to Angry Birds

This section presents each component of Figure 3 in detail by considering the domain of Angry Birds. Novelties in Angry Birds can appear in any game object. When explaining our difficulty measure formulation, we do not consider the novelties that appear in the birds' game object, as such novelties can usually be identified directly after a single shot by observing birds' behavior.

The first input is the initial state of the instance without any interaction. In our example domain, this is the game instance before shooting any birds. To represent the game scene, we use a 2D coordinate system where the x-axis is horizontal and oriented to the right while the y-axis is vertical and oriented to the top (Figure 4). P denotes all pixel points in a scene. For a pixel p_i, $x(p_i)$ and $y(p_i)$ denote its x and y coordinates. O represents all objects in the environment. Each object o_j (s.t. $o_j \in O$) comprises a set of pixels that can be mapped to a specific object (e.g. square wood).

The second input is the novelty present in the instance. In our example domain, novelties may appear in different object categories (i.e., wood, ice, stone, pigs) and the novel property could be any physics parameter (e.g. mass, friction, bounciness, etc). Thus, an example of the input is *(stone blocks, mass)*. These inputs are sent to the *target determining module* to search for possible target objects.

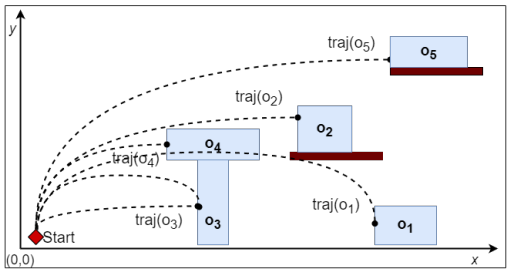

Figure 4: Representation of the object space. o_2, o_3, o_4 and o_5 satisfy the *left-of* relation to o_1. The trajectories to each object are denoted by the dotted line. A dot in the line represents a pixel point $p_i \in P$. o_2, o_3, o_4, and o_5 satisfy the *target* predicate. o_1 is not a target as the *traj(o_1)* intersects with o_4, which is in left to o_1. o_3 supports o_4. If o_3 moves, o_4 also moves: *impacted(o_3,o_4)* is true.

Target Determining Module

This module is used to identify the target objects. We consider the target objects as the objects that are directly reachable to interact. We do not consider platforms as target objects as they are static. We use the following predicates to determine the targets in our example domain.

- *left-of (o_i, o_j)*: if object o_j is in left of object o_i (Figure 4).
  left-of$(o_i, o_j) \equiv o_i \neq o_j \wedge x_{max}(o_i) > x_{min}(o_j)$,
  where: $x_{max}(o_i)$ and $x_{min}(o_j)$ are the maximum pixel coordinate of object o_i in *x* direction and minimum pixel coordinate of o_j in *x* direction respectively.
  $x_{max}(o_i)$ = *max*$(x(p_k) \; \forall \; p_k \in o_i)$,
  $x_{min}(o_j)$ = *min*$(x(p_k) \; \forall \; p_k \in o_j)$
- *traj(o_i)*: trajectory from a starting point to object o_i.
  As shown in Figure 4 for object o_3, a trajectory may contain multiple connections starting from a fixed position (slingshot in Angry Birds) to the connection point of the object. The connections can be represented using a set of points denoted by the dotted lines in Figure 4. We define:
  traj$(o_i) = \{t_1^i, t_2^i, ..., t_n^i\}$
  where, $t_k^i = \{p_{1k}, p_{2k}, ..., p_{nk}\}$
  i.e., a set of points that belong to one of the parabola trajectories and only $p_{n\cdot} \in o_i$.
- *target(o_i)*: if object o_i is a target object.
  o_i is a target if the object is directly reachable, i.e., there is at least one trajectory to o_i such that trajectory points do not intersect with any object with *left-of* relation to o_i according to our domain.
  target$(o_i) \equiv (\exists \; t^i \in traj(o_i)) \wedge t^i \notin o_j \; \forall \; \{o_j : left-of(o_i, o_j) \; \forall o_j \in O\}$

Similar to the above *left-of* relation, we can define *right-of, below,* or *above* relations according to the requirement in each domain. We can also define *traj(o_i)* and *target(o_i)* specific to each domain. For example, if the way to interact with the objects is to drop an object from above, *traj(o_i)* should be defined according to the path taken by the object in gravitational free fall and *target(o_i)* is determined by the trajectories that do not intersect with the objects in *above* relation to *target*$(o_i) \equiv (\exists \; t^i \in traj(o_i)) \wedge t^i \notin o_j \; \forall \; \{o_j : above(o_i, o_j) \; \forall o_j \in O\}$

Object Dynamics Reasoning module

After target objects are determined, it is necessary to identify the objects that can be moved due to the interactions with the target objects. This is achieved by using the *object movement analysis* component. We instantiate all components using our proposed qualitative physics algorithms. If the novel objects are among the impacted objects identified (defined below) or the target objects, the *detectability analysis* component captures if the novel objects move in a detectable way. We first define the following to aid the explanations of the methods used in the two components.

- *novel-object(o_i)*: if object o_i is a novel object. As defined earlier, o_i is a novel object if it violates the predefined mapping between appearance and physical parameters. i.e., object has changed physical parameter values.
- *impacted(o_i, o_j)*: if o_j is moved due to the interaction of a bird with the target object o_i. For example, if o_i supports o_j and o_i is hit by a bird, o_j also moves (See o_3 and o_4 in Figure 4. The reasoning for the identification of such objects is presented under *object movement analysis*).
- *detectable(o_i, o_j)*: if o_j moves in a detectable way due to the interaction of a bird with the target object o_i. detectable(o_i, o_j) returns true when o_j is a novel object and *impacted*(o_i, o_j) is true and o_j is impacted by the target object o_i in a detectable way. A case-based exploration of the detectability is conducted in *detectability analysis*.

Object Movement Analysis This section presents the qualitative physics approach used in identifying the objects that satisfy the *impacted* predicate presented above. i.e., we identify which objects move after an interaction with a target object. We use two algorithms 1) based on the stability, 2) based on the force propagation in the horizontal direction (Algorithm 1). We used the algorithm *vertical impact* proposed by Walega, Zawidzki, and Lechowski to reason about the stability of the objects. We also propose a new algorithm, *approximate horizontal influence* to check the impact on the objects located in the horizontal direction.

Vertical impact: This algorithm recursively checks the objects in a structure starting from the object that is directly impacted and returns a list of objects that may fall.

It exploits the rule which is the basis for stability investigation, "an object is stable if the vertical projection of the centre of mass of the object falls into the area of support base" (Zhang and Renz 2014). The algorithm contains eight steps where at each step object relationships are examined and substructures are constructed. The stability of objects is examined by approximating the center of mass of substructures and their supporters. A clear explanation of the algorithm is available in the work of Walega, Zawidzki, and Lechowski and is diagrammatically summarized in the extended version of our paper (Pinto et al. 2023). At the end of the eight steps, the algorithm returns the list of objects that may fall after the interaction with a target object.

Approximate horizontal influence: This algorithm examines the impact a target object can cause due to the force propagation on the objects located horizontally to the target.

We start by analysing if the target object can get destroyed due to the interaction. If it is not destroyed, we check if the

object will slide or it will flip as a result of the interaction (collision). Destruction of the target object heavily depends on the materials and the types of the two colliding objects and the velocity at the collision. In our example domain, we define the following predicate by considering the object materials (e.g., wood, ice, stone, pig) and the bird (e.g., red, blue, yellow). We approximate the velocity at the collision by using the law of energy conservation.

object-destroy$(o_i) \equiv o_i^{life} - damage < 0$. o_i^{life} is the object life and it depends on the material of the object and type of it (e.g. square wood-block, rectangular ice-block). This is a constant value for a specific object. *damage* depends on the type of the bird used and the relative velocity at collision. Damage caused by a bird type is a fixed value for a specific object, $o_i^{bird\_damage}$. *damage* can be approximated as $o_i^{bird\_damage} \times$ *relative-velocity* at collision. *relative-velocity* can be approximated using the law of energy conservation. Thus, the final formulae for the *object-destroy*(o_i) predicate can be given as, *object-destroy*$(o_i) \equiv (o_i^{life} - o_i^{bird\_damage} \sqrt{k1 \times (y_{start} - y_{target}) + k2_{bird}}) < 0$

where, $k1$ is an experimentally fixed constant value, and $k2_{bird}$ is a value based on the initial kinetic energy of the bird (In Angry Birds, the value only depends on the bird mass as the initial launch velocity is fixed because agents use the slingshot with full stretch). $(y_{start} - y_{target})$ is the height difference between slingshot and the target object.

If the *object-destroy*(o_i) predicate is false, we check the *object-flip*(o_i) predicate by considering object dimensions.

object-flip$(o_i) \equiv \frac{y_{max}(o_i) - y_{min}(o_i)}{x_{max}(o_i) - x_{min}(o_i)} > k_{flip}$,

where: $y_{max}(o_i)$ and $y_{min}(o_i)$ are the maximum pixel coordinate of object o_i in y direction and minimum pixel coordinate of o_j in y direction respectively. The k_{flip} is an experimentally fixed constant value.

k_{flip}= *flipping threshold*,
$y_{max}(o_i)$ = *max* $(y(p_j)\ \forall\ p_j \in o_i)$,
$y_{min}(o_i)$ = *min* $(y(p_j)\ \forall\ p_j \in o_i)$,
and $x_{max}(o_i)$, $x_{min}(o_i)$ are as defined previously.

These predicates hold the basis for the *approximate horizontal influence* algorithm. A pseudo-code of the process is demonstrated in Algorithm 1 and Figure 5 explains the terms *falling-arc*(o_i) and *sliding-path*(o_i) used in Algorithm 1.

- For a circle C, with centre $(x_{max}(o_i), y_{min}(o_i))$ and radius $(y_{max}(o_i) - y_{min}(o_i))$, let $q1$ be the set of pixel points in the first quadrant of C. *falling-arc*(o_i) returns the list of objects within the falling arc of object o_i (See Figure 5a). We define *falling-arc*(o_i) as follows:
  falling-arc$(o_i) \equiv \{o_j \in O \mid o_j \neq o_i \wedge (o_j \cap q1) \forall o_j \in O\}$
- *sliding-path*(o_i) returns the list of objects within the path the object o_i slides (See Figure 5b). We define *sliding-path*(o_i) as follows:
  sliding-path$(o_i) \equiv \{\ o_j \in O \mid o_i \neq o_j$
  $\wedge\ (x_{max}(o_i) < x_{min}(o_j) < x_{max}(o_i) + k_{sliding\_constant})$
  $\wedge\ ((y_{min}(o_i) < y_{max}(o_j) < y_{max}(o_i)) \vee (y_{min}(o_i) < y_{min}(o_j) < y_{max}(o_i)))\ \forall o_j \in O\}$
  where, $k_{sliding\_constant}$ is an experimentally determined distance that approximates the distance an object can slide after a collision.

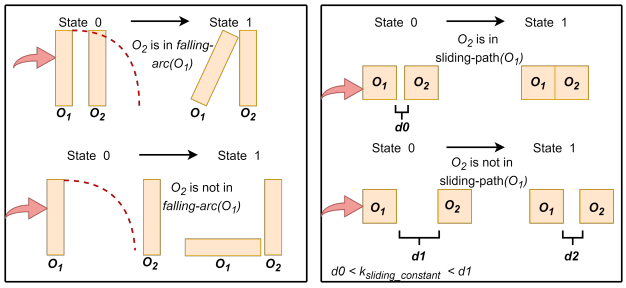

Figure 5: Left Figure (a) shows examples for *falling-arc*(o_1) and the right Figure (b) shows examples for *sliding-path*(o_1)

Algorithm 1: Approximate horizontal influence

Input: State representation of objects, target object (o_i)
Output: List of impacted objects

1: Initialize *horizontal-propagation(HP)_impact_list*
2: **if** $\neg$ (*object-destroy*(o_i)) **then**
3: **if** *object-flip*(o_i) **then**
4: *pending_list* = *falling-arc*(o_i)
5: **else**
6: *pending_list* = *sliding-path*(o_i)
7: **end if**
8: *closest_object* = $o_j \mid min(x_{min}(o_j) - x_{max}(o_i)\ \forall\ o_j \in pending\_list)$
9: Add *vertical impact*(*closest-object*) to *HP_impact_list*
10: **end if**
11: **return** *HP_impact_list*

In Algorithm 1 (line 8), we only limit to a single *closest_object* obtained from either the *falling-arc* or *sliding-path* according to the experimentation with our example domain. However, this can be altered according to the domain. The output of the *object movement analysis* module is the list of impacted objects obtained from the *vertical impact* algorithm and the *approximate horizontal influence* algorithm.

Detectability Analysis This section presents the case-based exploration in identifying the *detectable* predicate shown above. Once the set of impacted objects is available, we can categorize each object into at least one of the below cases that represent observable features in a physical world. The observable movement of the directly-hit object (i.e., target object) can be explained using the first three cases.

- Case 1: Directly hit and destroys
- Case 2: Directly hit and flips
- Case 3: Directly hit and slides

Apart from these three special cases, all objects subject to at least one of the following six cases. Case 4 and 5 focus on object rotation. We assumed that rotation of the impacted objects directly above and very close to static structures (ground or a platform) is hardly observable. Other objects could rotate due to the collisions with objects and there is a chance of observing the rotation when objects fall.

- Case 4: Falls from the top without rotating
- Case 5: Falls from the top while rotating

Case 6 and 7 focus on the objects that slide. The object may slide and stop, or it might fall if it's located above the

ground based on the sliding path.

- Case 6: Slide and stop
- Case 7: Slide and fall down

Case 8 and 9 focus on the objects which flip. Similar to the above two cases, it may either fall or stop based on location.

- Case 8: Flips and stop
- Case 9: Flips and fall down

The nine cases cover the majority of observable movements in Angry Birds (More details in (Pinto et al. 2023)).However, there could be situations that may be not captured using the nine cases. To evaluate if the novel object is detectable, we check if the object is moved in a detectable manner by considering the changed attribute along with the object type. Consider the following examples:

Example 1: Novelty in "friction" of stone blocks - If at least one impacted stone block satisfies the requirements for case 3, 6, or 7, we can detect the novelty (as friction changes can be observed when the object slides).

Example 2: Novelty in "bounciness" of wood objects - If at least one impacted wood object satisfies the requirements for case 2, 3, 4, 5, 6, 7, 8, or 9, we can detect the novelty (as bounciness can be observed when objects collide).

The output of this module enables to capture the objects that satisfy *detectable* predicate for each target object.

Difficulty Computation Module

This component quantifies the difficulty of detecting novelty for each game instance. We propose two algorithms to calculate the detection difficulty. Factors including the novelty in the object, the placement of the objects, the number of detectable objects, the number of reachable objects, and the number of interactions available (number of birds in Angry Birds) are considered when developing both methods.

We define the following to identify the most influential target object to interact with (i.e., the target object that gives the most information about objects movements. We refer to this as the *best-target*).

- *impact-score(o_i)*: The heuristic impact score of *target(o_i)* is defined based on the objects moved and the novelty.

  Example 1: If the novelty is in only one object in the instance, the *score* per each object moved = 1

  Example 2: If the novelty is in objects with the same material (wood, ice, stone), the *score* per material moved=1

  Example 3: If the novelty is in object types and if the player is informed that the wood objects are not novel, the *score* per each wood object moved = 0, the *score* per other types of objects moved = 1

  impact-score(o_i) = $\sum_{o_j \in O | impacted(o_i, o_j)} score_{o_j}$

- *best-target*: The target object with the highest *impact-score*. If there are multiple objects with the same impact-score, the first object from all objects is selected.

  best-target $\equiv$ o_i | *target(o_i)* $\wedge$ *impact-score(o_i)* = *max(impact-score(o_i))* $\forall$ $o_i \in O$

Algorithm 2: Probabilistic interaction difficulty

Input: State representation of objects *(O)* **Output**: *PID*

```
Initialize PID = 0
for i in total_number_of_interactions do
    N_i = | { o_j ∈ O | target(o_j) ∀ o_j ∈ O } |
    n_i = | { o_j ∈ O | (target(o_j) ∧ ∃ o_k ∈ O s.t. novel-object(o_k) ∧ detectable(o_j, o_k)) ∀ o_j, o_k ∈ O } |
    M_i = (N_i − n_i) / N_i
    PID += M_i
    if M_i ≠ 1 then
        break
    else
        Shoot at the best-target
        Update state of objects
    end if
end for
PID = PID / total_number_of_interactions
return PID
```

Probabilistic interaction difficulty (PID) Algorithm 2 is based on the intuition that the probability of novelty detection depends on the number of novel objects available. Intuitively, if the probability of finding a target that impacts the novel object in a detectable way is lower, the difficulty is higher. *PID* is initialized at zero, and the algorithm loops over the number of possible interactions (i.e., number of birds available in Angry Birds) while updating the *PID*. To proceed to the next interaction, it is assumed that the agent shoots the best-target and the objects in the environment are updated along with the search space (which objects to explore next). The terms, N_i is the total number of target objects and n_i represents the total number of target objects which makes the novel object move in a detectable way in the given state. Thus, M_i is the proportion of targets that do not yield a detectable movement. At the end of the computation, *PID* is normalized to [0,1] (1 indicates the highest difficulty, and *PID* is unitless). One limitation of this algorithm is that it only considers the *best-target* when updating the next state instead of considering all possible targets. This is due to time restrictions and works under the assumption that an intelligent agent would always select the *best-target*.

Best-shot interaction difficulty (BID) Algorithm 3 is inspired by an intelligent human-like agent and is based on the interaction which yields the most information. Here we try to maximize the chance of novelty detection by making the most influential interaction (i.e., always shooting at the *best-target:* o_k*). The algorithm loops over the number of possible interactions that can be made: if the novelty is undetectable by shooting at the best-target, it proceeds after updating the environment, the search space (which objects to explore next), and *BID*. Similar to Algorithm 2, *BID* is normalized to [0,1], where 1 indicates the highest difficulty and is unitless.

These two difficulty algorithms can be used separately or collectively according to the suitability of the study.

5 Experimental Evaluation

As the difficulty measures we proposed is general, we examined the relationship between our proposed difficulty measure and human perception. We conducted an experiment approved by the Australian National University hu-

Algorithm 3: Best-shot based interaction difficulty

Input: State representation of objects **Output**: *BID*

Initialize *BID = 0*
Initialize *detection_flag = False*
for i in *total_number_of_interactions* **do**
    BID = BID + 1
    if $(\exists o_j \in O \mid novel - object(o_j) \wedge detectable(o_k*, o_j))$ ***then***
        detection_flag = True
        break
    else
        Shoot at the best-target
        Update state of objects
    end if
end for
if *detection_flag = False* **then**
    BID = total_number_of_interactions + 1
end if
BID = (BID - 1) / total_number_of_interactions
return *BID*

man ethics committee (protocol-2020/717). We gathered data from 20 voluntary participants (aged 20-35, including males and females) with no prior knowledge of the tested novelties. Participants played 10 instances without novelty (generated from Angry Birds levels generator (Stephenson and Renz 2017)) to familiarize themselves with the game physics and dynamics. Then, they played 15 instances, each featuring one of three different novelties. We measured the difficulty of detecting each novelty using our proposed approach. Each participant was allowed to play the novel instance only once to detect if there is any novelty in the game objects. If the novelty was detected, we recorded the number of interactions (number of shots) the participant used to detect that novelty. We requested the participant to provide a simple description of the observation to validate the results. Each participant took approximately 40-50 minutes to complete the experiment. The novelties we generated are:

- **Type 1 (T1):** The parameter *gravity scale* of pigs is decreased twice the original value. Pigs fall down slower due to this novelty.
- **Type 2 (T2):** The parameter *bounciness* of wood objects is increased by four times the original value. This makes the wood objects bouncier.
- **Type 3 (T3):** The parameter *life* of stone objects increased by five times. This makes stone blocks difficult to destroy.

Game Instance Selection A set of 100 game instances was generated from the state-of-the-art (SOTA) level generator (Stephenson and Renz 2017) and the novelty game instances were created for each novelty type. We then computed difficulty using the two algorithms, *PID* and *BID* for each instance. We combined the two values: *Difficulty Value* = α*PID* + *(1-*α*)BID*, where $\alpha \in [0,1]$, can be adjusted based on the importance of the two algorithms in an experiment. In our experiment, we got $\alpha = 0.5$ to give equal importance. Game instances within each novelty type were then classified into three categories: *easy (e), medium (m), hard (h)*. Game instances with values $<$ the value at 33.33% percentile, 33.33% - 66.67%, and values $>$ 66.67% were considered as *e, m*, and *h* instances respectively. The game

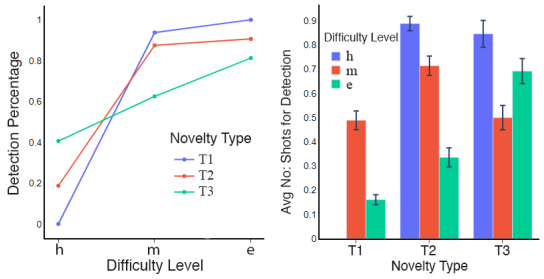

Figure 6: Experiment results from human participants. The left figure (a) shows the percentage of novelty detection and the right figure (b) shows the average normalized number of shots for novelty detection for each difficulty level. Error bars represent the standard error. *e,m,h* indicate easy, medium, and hard categories.

instances used for the experiment were selected randomly from each category. However, techniques such as harmonic mean/clustering methods could also be utilized to categorize based on the data available.

Results According to our difficulty measure, we expect the percentage of novelty detection to decrease in the order *e, m*, and *h* (Algorithm 2). Ideally, if the novelty is detected, we expect a lower number of interactions to detect the novelty in the category *e* and a higher number of interactions in the category *h* (Algorithm 3). Figure 6a illustrates the percentage of human participants who correctly detected the novelty for each novelty type in the three difficulty levels. In line with our hypothesis, the lowest percentage of detection is recorded in the category *h* and the highest is recorded in the category *e*. This observation is consistent for all three experimented novelty types. For the T1 novelty type, none of the participants were able to detect the novelty in the category *h*, while all the participants detected it in category *e*.

Figure 6b summarizes the average normalized number of shots needed for detection for each difficulty level for the three novelty types. That is, for each participant, the number of shots taken for detection is normalized by the total number of possible interactions (i.e., the number of birds in the game instance). For novelty type T1, the category *h* is not presented as none of the participants detected the T1 novelty type. The *m* and *e* categories follow our expectation by producing a lower value for the category *e*. Similarly, T2 results are also consistent with our expectation. For T3, while the category *h* gives the highest normalized interactions for detection, the category *m* is lower than the category *e*. According to our observations, some participants used more shots to confirm that stone-blocks have a higher health value even though they already detected this novelty earlier and some participants did not notice the change in stone-blocks at all. Overall, the difficulty of novelty detection for human participants falls in line with the calculated difficulty values.

6 Discussion and Conclusion

Detecting novelty is an important capability for an intelligent system in an open-world environment. In real-world situations, an agent needs to reason about physics in order to detect novel objects with different physical parameters. These novelties often vary in their difficulty of detection and have not been studied before this paper. However, un-

derstanding this difficulty can be an important aspect of conducting a robust and fair evaluation. Thus, we have proposed a method to quantify the difficulty of novelty detection using qualitative physics. Our method is agent-independent and can be used to make more accurate conclusions about the detection capabilities of different agents. This measure was applied in the Angry Birds domain, and validated by comparing the results of the proposed measure with the performance of human participants. To define the physical reasoning predicates, we have used quantitative thresholds based on domain knowledge.

The different components and algorithms that were introduced in this paper can also be applied to other research problems. When formulating our novelty difficulty measure, we proposed the algorithm *approximate horizontal influence* that could also be used as a component for agents to predict the influence of moving an object. This is an improvement to the prior work (Zhang and Renz 2014; Walega, Zawidzki, and Lechowski 2016) as it considers objects that are disconnected in the horizontal direction. Our difficulty formulation can also be used to create novel game instances at a predefined difficulty of novelty detection. It can be used as a component in the SOTA novelty generation framework for Angry Birds (Gamage et al. 2021) to generate novel game instances with a predefined difficulty. This facilitates research in open-world learning agent development by creating different instances with different levels of difficulty.

We plan to extend our study to address limitations such as generalizing our presented qualitative reasoning algorithms in *object movement analysis* to other domains. Moreover, we have discussed how the difficulty formulation can be applied to PHYRE (Bakhtin et al. 2019) in the extended version of this paper (Pinto et al. 2023) and we plan to extend the framework to suit a wider variety of novelties and be applicable to a wider range of domains. In this paper, we laid a foundation for quantifying the difficulty of novelty detection that aids to conduct a sound open-world evaluation.

Acknowledgments

This research was sponsored by the Defense Advanced Research Projects Agency (DARPA) SAIL-ON program.

References

Allen, J. F. 1983. Maintaining Knowledge about Temporal Intervals. *Commun. ACM*, 26(11): 832–843.

Bakhtin, A.; van der Maaten, L.; Johnson, J.; Gustafson, L.; and Girshick, R. B. 2019. PHYRE: A New Benchmark for Physical Reasoning. In *NeurIPS*.

Boult, T.; Grabowicz, P. A.; Prijatelj, D.; Stern, R.; Holder, L.; Alspector, J.; Jafarzadeh, M.; Ahmad, T.; Dhamija, A. R.; Cli; Cruz, S.; Shrivastava, A.; Vondrick, C.; and Scheirer, W. 2021. A Unifying Framework for Formal Theories of Novelty: Framework, Examples and Discussion. In *AAAI*.

Chollet, F. 2019. On the Measure of Intelligence. *arXiv e-prints*, arXiv:1911.01547.

Ferreira, L.; and Toledo, C. 2014. A search-based approach for generating Angry Birds levels. In *2014 IEEE Conference on Computational Intelligence and Games*, 1–8.

Franco, J. P.; Yadav, N.; Bossaerts, P.; and Murawski, C. 2018. Where the really hard choices are: A general framework to quantify decision difficulty. *bioRxiv*.

Gamage, C.; Pinto, V.; Xue, C.; Stephenson, M.; Zhang, P.; and Renz, J. 2021. Novelty Generation Framework for AI Agents in Angry Birds Style Physics Games. In *COG*.

Ge, X.; Renz, J.; and Zhang, P. 2016. Visual Detection of Unknown Objects in Video Games Using Qualitative Stability Analysis. *IEEE Transactions on Computational Intelligence and AI in Games*, 8(2): 166–177.

Giesel, M.; and Gegenfurtner, K. R. 2010. Color appearance of real objects varying in material, hue, and shape. *Journal of Vision*, 10(9): 10–10.

Langley, P. 2020. Open-World Learning for Radically Autonomous Agents. In *AAAI*.

Laubert, C.; and Parlamis, J. 2019. Are You Angry (Happy, Sad) or Aren't You? Emotion Detection Difficulty in Email Negotiation. *Group Decision and Negotiation*, 28: 377–413.

Perner, P. 2018. Determining the Similarity Between Two Arbitrary 2D Shapes and Its Application to Biological Objects. *IJCSE*.

Pinto, V.; Xue, C.; Gamage, C. N.; Stephenson, M.; and Renz, J. 2023. The Difficulty of Novelty Detection in Open-World Physical Domains: An Application to Angry Birds. *arXiv: 2106.08670*.

Renz, J.; Ge, X.; Gould, S.; and Zhang, P. 2015. The Angry Birds AI Competition. *AI Magazine*, 36: 85–87.

Renz, J.; Ge, X.; Stephenson, M.; and Zhang, P. 2019. AI meets Angry Birds. *Nature Machine Intelligence*, 1.

SAIL-ON-BBA. 2019. Broad Agency Announcement, Science of Artificial Intelligence and Learning for Open-world Novelty (SAIL-ON).

Senator, T. 2019. Science of Artificial Intelligence and Learning for Open-world Novelty (SAIL-ON).

Stephenson, M.; and Renz, J. 2017. Generating varied, stable and solvable levels for Angry Birds style physics games. In *2017 IEEE Conference on Computational Intelligence and Games (CIG)*, 288–295.

Stephenson, M.; Renz, J.; Ge, X.; Ferreira, L.; Togelius, J.; and Zhang, P. 2019. The 2017 AIBirds Level Generation Competition. *IEEE Transactions on Games*, 275–284.

Takatalo, J.; Häkkinen, J.; Kaistinen, J.; and Nyman, G. 2010. Presence, Involvement, and Flow in Digital Games. *Evaluating User Experience in Games*, 23–46.

Walega, P. A.; Zawidzki, M.; and Lechowski, T. 2016. Qualitative Physics in Angry Birds. *IEEE Transactions on Computational Intelligence and AI in Games*, 8(02): 152–165.

Yom-Tov, E.; Fine, S.; Carmel, D.; and Darlow, A. 2005. Learning to estimate query difficulty: Including applications to missing content detection and distributed information retrieval. *Proc. ACM SIGIR*, 512–519.

Zhang, P.; and Renz, J. 2014. Qualitative Spatial Representation and Reasoning in Angry Birds: The Extended Rectangle Algebra. In *Proceedings of the Fourteenth International Conference on Principles of Knowledge Representation and Reasoning*, KR'14, 378–387. AAAI Press.