EPRIDE Report

Xin Lu (u7165475), Sihan Lin (u6569267), Lachlan Stewart (u7284324)

May 30, 2022

Link to our Gitlab repo: `https://gitlab.cecs.anu.edu.au/u7165475/2500-rl-benchmark.git`

1 Abstract

Existing intuitive physical reasoning and reinforcement learning benchmarks do not involve embodied agents in physically dynamic environments. This is an important issue which we sought to address through the creation of a novel benchmark. EPRIDE (Embodied Physical Reasoning In Dynamic Environments) is an organised set of physical reasoning puzzles in 2D which requires an embodied agent with lasting control to reach the top of its environment. EPRIDE is designed to be accessible to reinforcement learning, in-particular. In this report we discuss our motivations for creating EPRIDE, challenges we faced, and an analysis of our benchmark tasks.

2 Our research question and motivations (Q1/Q2):

Because we are benchmark-oriented, we believe our research questions are slightly different in nature to most approach-oriented AI research. To that end, we leave out questions such as which models and approaches perform the best.

2.1 What is an embodied physical reasoning task?

Duan et al. (2022) discusses how intuitive physics tasks can be further extended by requiring agents to perform action tasks based on their current capacities to infer mass and friction. However, the meaning of action tasks is vague and open to interpretation, which cannot serve as a building ground of a benchmark.

Following this discussion, we identified embodiment and lasting actions as key features which would make benchmark tasks more challenging than existing benchmarks in PHYRE and PhyQ.

We then defined embodied physical reasoning to be an action-oriented decision-making process which is primarily dependent on the complex relationships between the latent physical properties of objects, including the agent's embodied form. Latent physical properties include geometry, position, motion, mass, friction, gravity, etc.

2.2 Do existing benchmarks address all important aspects of physical reasoning? If not, what is missing?

After reading a cross-section of relevant literature on physical reasoning benchmarks and reinforcement learning benchmarks (Renz. et al., 2021) (Bakhtin et al., 2019), we developed the opinion that existing benchmarks are indeed missing something.

In existing benchmarks for *physical reasoning*, agents do not physically manifest in their environment as an embodied form that they have lasting and continuous control over. When AI systems are applied in critical physical reasoning scenarios, such as driving a car in the real world, we will want to know that they were thoroughly tested on a benchmark for embodied physical reasoning.

At the same time, existing *reinforcement learning* benchmarks, which often involve an embodied agent, do not require the agent to perform any physical reasoning task. Though many locomotion-based benchmarks simulate physical dynamics for the agent's embodied form, the environment surrounding the agent is often static. For example, we see this in the Deepmind Control Suite (Tassa et al., 2018).

2.3 How do we create a good benchmark?

We want to create a benchmark that can attract future research. To achieve this, we believe that our benchmark needs to satisfy the following:

- Relevance
- Ease of use
- Robustness
- Appropriate difficulty

As previously discussed, we believe that our benchmark is highly relevant both for its novelty and its comparability to the real world.

We formulated our benchmark templates as reinforcement learning environments using tensorflow, which streamlined the template creation process to a matter of minutes. Future researchers would have this available to them as well.

Benchmark robustness is a nuanced topic. Often in machine learning the robustness of a benchmark is predicated on the quality of its underlying dataset. Our benchmark does not surround a finite pre-determined dataset, which makes this question different for us. We identified determinism and variation in our template tasks as methods that we could apply to build a more robust benchmark.

The difficulty of our benchmark tasks for AI is an open research question. As the humans who designed the benchmark, we have had the opportunity to test-drive each template, and we know that our benchmark is well-within human capabilities. We will provide more insight into the difficulty of our benchmark tasks through our analysis later in this report.

2.4 How can we ensure that our benchmark addresses embodied physical reasoning?

Benchmarks exist to measure the magnitude of research progress in a particular direction. We wish to make it clear that our benchmark is concerned with physical reasoning.

We demonstrate in our analysis that some agents which do not perform physical reasoning do not perform well, which suggests in the contrapositive that agents which do perform well will perform physical reasoning.

We also argue that the similarities between our artificial benchmark and the real-world, and the artificial process of reinforcement learning and natural evolution, are significant. The latter comparison has been made by highly influential researchers in the field:

"The theory of reinforcement learning provides a normative account, deeply rooted in psychological and neuroscientific perspectives on animal behaviour, of how agents may optimize their control of an environment." (Minh. V, et al., 2015)

Our benchmark also demands 'pushing' and 'avoiding' behaviours from successful agents. There is no question that the ability to perform such actions in appropriate circumstances predicates the survival of human beings.

We would also draw attention to the nature of perception in human beings. It is believed that human beings infer the meaning and utility of objects as a necessary step to classify them and take subsequent actions, not the other way around (Peterson, 2022). We suggest that if the similarities we have highlighted are significant, then there is reason to suggest that reinforcement learning on our benchmark will lead agents to do the same.

3 Current State-of-the-Art (Q3):

Our research surrounds the creation of a novel benchmark. A research team attempting a new approach to a machine learning problem would compare their model's performance to the performance of models resulting from previous attempts at the problem, we compare our benchmark to existing benchmarks. Our criterion for a benchmark which is state-of-the-art is that the research community is *presently engaged* with it. This is because to incite research, we believe that a benchmark must be relevant, easy to interact with, well-tested, and difficult enough to inspire innovation, without being prohibitively complex. The benchmarks we consider to be state-of-the-art are facilitating ongoing research today and exemplify these qualities.

We discuss some tactics that we have seen in the literature for endowing benchmarks with these qualities.

3.1 Relevance

Authors of novel benchmarks should justify the relevance of their benchmark by stating similarities and differences to existing benchmarks and by discussing the implications of high performance on the benchmark.

To provide an entryway into the benchmark, authors should explain where the benchmark sits in relation to existing benchmarks. This directs research towards approaches which perform well on similar benchmarks, and away from those which are known not to perform well.

A reward for effort is the most fundamental form of incentive. Benchmarks inherently provide this by serving as a metric which can confirm steady technological progress, but more can be done. Authors should strive to justify a statement such as the following:
"if an approach can perform to X level on this benchmark, then Y can be done", where Y is some meaningful application of the technology making use of the new approach.

3.2 Ease of use

Benchmarks should be easy to use because this maximises the time that can be dedicated to developing a new approach. They should also be optimized. A benchmark which takes many hours to evaluate a model's performance makes it drastically harder to make fine adjustments such as tuning hyperparameters. The comprehensiveness of a benchmark can impact how long it takes to evaluate a model, so another tactic is to break down the benchmark into manageable tasks, tiers, or similar.

3.3 Robustness

We are aware of cautionary tales about datasets which are known to have been tainted with confounding patterns. It is compelling to read when authors have documented focused efforts towards identifying and eliminating these, improving the robustness of their benchmark.

3.4 Difficulty

Benchmark tasks should sufficiently challenge agents to achieve goals, while still allowing good agents to perform well. More importantly, significant performance differences between baseline agents and sophisticated learning agents indicate the quality of benchmark task design, because such differences show the close relations between success in benchmark tasks and learning capacities of agents.

4 Papers detailing the state of the art (Q3):

As previously discussed, our benchmark exists at the intersection between reinforcement learning and intuitive physical reasoning. We discuss existing benchmarks in each area.

4.1 Phyre: A new benchmark for physical reasoning - Sihan Lin (u6569267):

Bakhtin et al. (2019) proposed a new benchmark *PHYRE* to evaluate the learning algorithm's ability to solve physical reasoning puzzles. The benchmark *PHYRE* has a boxed 2D environment that contains static bodies (which cannot be moved) and dynamic objects (which will be affected by gravity and can interact with other dynamic objects), and different types of objects can be distinguished by colour. In the initial state of the task, the agent can perform a single action, which is placing one or more dynamic bodies into any valid position in the environment. After the agent performs the action, the environment will simulate the future physical interaction frames, and the agent will not be able to perform any more actions in the process. The goal of tasks is defined as a relationship between two bodies that should be achieved in the final state, and the current benchmark only contains one type of relationship: make two bodies touch each other for at least 3 seconds.

There are two types of benchmark tiers, and different tiers have different action sets:

- **PHYRE-B:** only place one single ball into the environment
- **PHYRE-2B:** place two balls into the environment

Besides the action set, different benchmark tiers will also have a different set of puzzles. In the PHYRE, each tier will have 25 unique templates, and each template contains 100 similar variant puzzle tasks. This design not only encourages the agent to learn the underlying generalized physics reasoning rules but also allows different task variants to be used in the training and testing process to improve the robustness of the benchmark.

Further, each benchmark tier also designed two settings to measure the generalization ability between agents:

- Within template: train & test with different task variants in the same template.
- Cross-template: train & test with different templates.

The research also conducted experiments to collect the baseline results of the benchmark. Five based agents have been used in the experiment, including:

1. Random agent (RAND): Purely random action in the action space without training.
2. Non-parametric agent (MEM): Randomly generate actions in the training stage, then reproduce actions that are most likely to solve the task in the testing stage.
3. Non-parametric agent with online learning (MEM-O): Same to the MEM but the agent will continue to learn (collecting reward of actions) in the testing stage.
4. Deep Q-network agent (DQN): Use a convolutional network (CNN) and Deep Q-network to train the agent with observation-action-reward pairs, and use a fusion module to make the prediction from observation.
5. Deep Q-network agent with online learning (DQN-O): Same to the DQN agent, but the agent will still update rewards in the testing stage by performing gradient descent.

The experiment places different agents in the same set of deterministic testing and training environments and measures the performance of agents by AUCCESS (task complete rate vs the number of attempts). Their experiment found that the online agent is more efficient in PHYRE, and illustrated that PHYRE-2B tier in cross-template settings is much more difficult for all agents.

4.2 A survey of benchmarking frameworks for reinforcement learning, 2020 - Lachlan Stewart (u7284324):

This paper explores the landscape of benchmarking for reinforcement learning via the examination of six influential benchmarks, which we too believe are state-of-the-art. After summarizing key RL concepts and challenges, the paper draws attention to the importance of benchmarking as well as the wide range of problems which these benchmarks encompass. This paper also examines significant trends in the rapid development of new benchmarks such as complexity, accessibility and reproducibility.

OpenAI Gym, published in 2016, serves to unite a wide range of reinforcement learning environments as a suite. As of 2022, OpenAI gym's suite of environments contains the environments of RLLab and ALE. This paper praises the OpenAI gym benchmark for its diverse range of environments and the provision of tools for the standardized comparison of algorithms and the reporting of results in literature. Based on our observations, as well as the summary provided by this paper, new benchmarks should be developed to emulate the features of OpenAI gym so that they can be added to its catalogue, providing immediate access to the research community.

The ALE benchmark, published in 2013, is highlighted as a benchmark which introduced a range of environments with challenging reward structures. This benchmark saw the development of Deep-Q Learning, a seminal advancement in reinforcement learning. Today, research continues to be challenged by the sparse reward structures and vast state spaces of games in the ALE benchmark such as Montezuma's Revenge.

RLLab is noted for pioneering continuous control variants of locomotion and classic control environments. They provide a set of 31 environments which are tiered by difficulty. Environments are made more difficult by limiting the quality of observational input. For instance, by completely removing parts of the observation, adding noise to the observation, or by adding a temporal delay to the observation. We, as a team, find these considerations inspiring.

Robocup Keepaway Soccer, published in 1997 is a benchmark for evaluating machine learning algorithms. It formed the basis for several international competitions. This paper praises the level of complexity of the environment for being "simple enough to be solved successfully, but complex enough that straightforward solutions are not sufficient".

Microsoft Textworld, published in 2018, is a text-based reinforcement learning benchmark which presents the challenges of common-sense reasoning, long-term planning, and exploration. This paper makes note of the efforts of the developers to add generative functionality to the benchmark, and that this will be of great use to future research.

In summary, this paper provides a comprehensive overview of current influential reinforcement learning benchmarks. Challenges which set each benchmark apart are identified, as well as the individual contributions each benchmark makes to the field. The paper calls for the research community to strive to maintain standards and integrity. After reading this paper, one can feel convinced that benchmarks are a primary force driving AI research. Perhaps more time should be spent developing them, instead of pursuing high performance on what already exists, so that the boundaries can be pushed in a more targeted and efficient manner.

4.3 PhyQ: A Benchmark for Physical Reasoning - Xin Lu (u7165475):

4.3.1 Benchmark design

PhyQ tasks enable agents to apply acquired knowledge when testing their generalization ability.

- **PhyQ guarantees one physical scenario will be solved by one physical rule:** PHYRE tasks can be solved by multiple physical rules and thus it does not necessarily imply the knowledge learned by agents in training will be tested in within-template testing or cross-templates testing. However, PhyQ's "one physical scenario, one physical rule" design methodology ensures that an agent will be tested on its ability to generalize the learned knowledge locally or broadly.

PhyQ tasks focus on evaluating agents' local generalization and broad generalization ability. PhyQ contains multiple scenarios that examine multiple physical rules which are inspired by physical reasoning abilities of human infancy, and by real world robotics application demands. In every scenario, different subsets of task templates are used in training and testing. Each task template is constructed using a different environment setting compared to other task templates. Within the same task template, PhyQ provides 100 tasks that have similar environment settings but with different generated dynamics. Also, a task variation factor is provided to future researchers for generating more tasks in the same task template by their needs.

- **Local generalization:** Within the same task template, PhyQ examines the agent's local generalization ability by training it on a subset of tasks and then testing it on a disjoint subset of tasks.
- **Broad generalization:** Within the same scenario, PhyQ examines the agent's broad generalization ability by training it on a subset of task templates and then testing it on a disjoint subset of task templates. The testing task templates contain unseen environment settings but will essentially require the agent to apply the same physical rule it learned during training to solve the testing task templates.

4.3.2 Baseline agents and PhyQ score

PhyQ score is used to evaluate local and broad generalization performance, which is calculated as an agent's average pass rate in the testing task template or testing scenario. Researchers of PhyQ also compared the performance of human players, deep learning agents with different inputs and learning strategies, heuristic agents, and random agents in their paper.

4.3.3 Experiment Outcomes

- Human players generally outperform artificial agents in both local and broad generalizations.
- **Local generalization:** Deep learning agents outperform heuristic agents, which is as expected, since there are large amounts of dense data that serve as inputs to learning agents.
- **Broad generalization:** Heuristic agents perform well in testing scenarios that are aligned with their built-in strategies, and heuristic agents generally outperform learning agents in broad generalization. This result shows current deep learning agents still struggle to generalize its learned knowledge from one environment to other unseen environments. It also raises the concern that current deep learning methods tend to learn more about the statistical pattern rather than the underlying physical rules from training.

5 The design and development of our benchmark (Q4):

5.1 Studying PHYRE:

By studying the PHYRE benchmark, we identified approaches that we could adapt and noted the following:

- Structuring the benchmark with templates and tasks is well organized and facilitates the systematic study of generalization.
- Restricting the environment to two dimensions makes implementation and testing simpler and makes goals and observational input clearer to define.
- A small range of different object types allows for a wide range of potential templates.
- A baseline should be identified to serve as an incentive for future attempts to solve the benchmark.
- The goal should be consistent across all tasks.
- An interactive version of the benchmark is useful for establishing the robustness of tasks.

5.2 Benchmark task concept iterations:

We had a few initial concepts before settling on our final design.

In keeping with the similarities to a 'video-game', an early idea was to create an Asteroids clone, where the agent controlled a spaceship and the Asteroids exhibited rigid body physics upon collision. We decided that this did not address enough aspects of physical reasoning to be worthwhile as a benchmark, as gravity and friction were missing. This was a step in the right direction. We decided that it was important to be able to test the full range of physical concepts and it introduced the idea of the agent avoiding objects in pursuit of a goal.

We also came up with another idea which allowed the agent to interact intentionally with dynamic objects in the environment (i.e. pushing or throwing objects). The goal would have been to move particular objects in the environment to a particular location. We also abandoned this idea because we believed this expanded action space caused too many complications.

We settled on a 'run-the-gauntlet'-style set of tasks. The agent's goal is to reach the top, from the bottom, but there may be obstacles or puzzles in its way. We implemented a demo template to prove the concept. This template is named "Seesaw", in which two red objects of differing mass rock a seesaw so that only one path is unobstructed. We imagined that if an agent could understand that a massive object unbalances a less massive object, then it could choose the correct path consistently, no matter where the two objects were originally placed. This template idea showed us that the new concept for the benchmark facilitated designs which can measure a more meaningful range of physical concepts. It takes care, creativity, and time to think of these designs.

5.3 Benchmark Task Details:

We first identified the avoiding task because it requires embodied agents to perform actions in all timestamp. We then compared this type of task to the current PHYRE task and proposed the need to design tasks that also require agents to actively interact with, and influence physically dynamic objects in the testing environment. Because agents in avoiding tasks are mostly expected to succeed by understanding the physical dynamics of other objects via observation, but not by actively interacting with them.

Based on two types of tasks and the need of a coherent testing environment, we finally came up with the idea of combining avoiding and interacting as the main structure when designing our templates.

We opted to have three kinds of objects in the environment. **Static (black)** objects, which are immovable, and two kinds of dynamic objects subject to gravity, friction, and collision. **Safe (blue)** objects, which the agent can safely contact, and **Danger (red)** objects, which the agent must avoid direct contact with. The agent (green), is a small circular body with enough mass and speed to push around small to medium-sized bodies.

The environment is bounded within a 20x40 box enclosed by static walls, and we stipulate that the path to the goal should never bear resemblance to a maze defined by static walls. This is because we want our benchmark to focus on physical reasoning, and not path-finding. We use Box2D to simulate the physics step, which is a deterministic physics engine. The same policy will always achieve the same result for the same variant.

5.4 Formulating our benchmark templates as classic reinforcement learning environments - observations and rewards:

We found that our benchmark templates highly resembled classic reinforcement learning environments. We studied the environments in OpenAI-gym's open-source collection, which includes Atari games and locomotion problems. This taught us to rigorously define our reward and observation functions.

This was not a trivial task. There were many different ideas within the group about how to approach the reward and the observation and it was difficult to justify one over another. We discuss some of the different approaches.

5.5 Observation:

5.5.1 Image Input:

We firstly started from the PHYRE's image input approach, and designed an image-based observation. In the first implementation, our agent takes a 256x256 pixel image vector of the entire scenario as the observation. However, we found that this size of input is too large for our device, which requires more than 16GB of memory to train a basic Deep Q Network agent. Therefore, we decided to scale down the image vector to 40x80 pixels, which successfully reduced the observation size to an acceptable level.

Further, compared to the PHYRE which uses 7 channels for each pixel to indicate the type of object, our implementation uses normalized RGB color value for each pixel due to the performance issue: it takes much longer than we expected to perform point testing. However, it also means that the agents will have to infer the object type in the pixel from the RGB value in the image observation, which could be challenging.

In addition, unlike PHYRE where the model only needs to make the decision in the first frame, the agents in our benchmark have to perform actions over all timestamps. As a result, the single frame image observation may not be enough for the agent to make rational decisions, because the information about object velocity cannot be observed in the single image. Therefore, we introduced additional last two frame image vectors in the observation, and hopefully it will allow the agent to infer the object velocity from the frame difference in the observation.

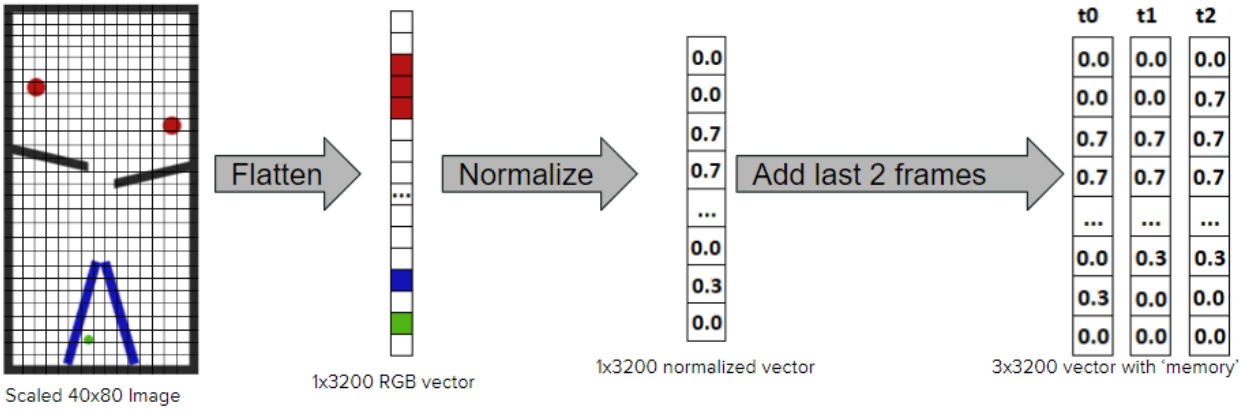


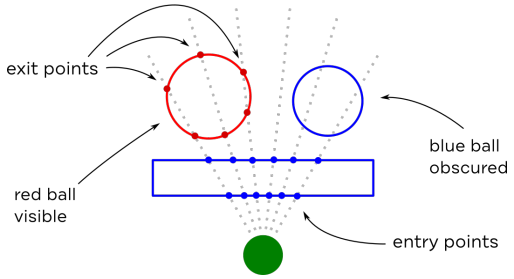
Figure 1: generate image observation from the state

The strength of this approach, in theory, is that the image provides comprehensive information about the entire environment at a homogenous resolution. This means the agent has information available to it about objects that are distanced, which might facilitate long-term planning.

5.5.2 Raycasting input:

The approach here was to localize the agent's observation to its embodied form. This was done through Box2D's raycasting faculties. Rays are cast at equal angular offsets and collide with the other bodies in the environment. The distance along the ray, as a fraction of the maximum length of the ray, as well as the angle of the surface normal of the edge intersected, is added to the input vector. A ray may enter and exit the body, which are treated as two separate intersections. This way, the agent sees the full geometry of a body.

This process is carried out for each of the three kinds of objects in the environment separately. This means that if a blue object is in front of a red object, for example, both can still be seen, but if a blue object is in front of another blue object, then the one behind is obscured from the agent's perception.



For 100 rays, the total input size is:
$100 \cdot 3 \cdot 4 = 1,200$.

In theory, this observation prioritizes the resolution of proximal information and reduces the input size.

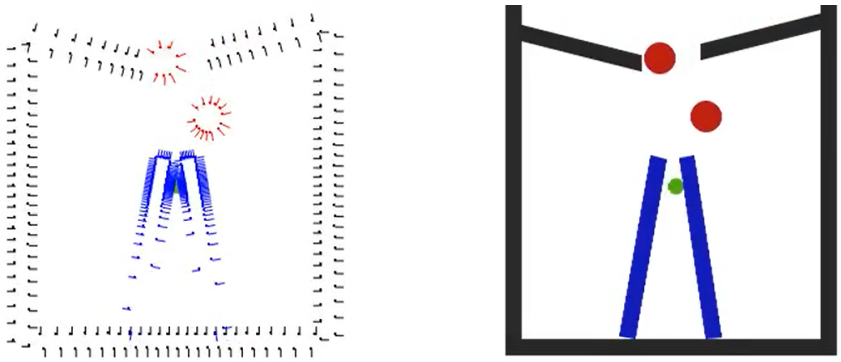


Figure 2: Rendering of raycasting information vs normal rendering. Important details of objects are fairly preserved

5.6 Reward

Designing effective rewards for reinforcement learning is an ongoing research issue and one we contend with regularly.
We identified two areas of concern when it came to our approaches:

- Sparseness
- Alignment with the goal of reaching the top

In practice, to our surprise, it is not the case that rewards should be directly proportional to progress towards the goal. A good example of this is the mountain-car problem (Moore. A, 1990). In this environment, the agent has two actions (left and right), two inputs, x-coordinate and velocity, and is governed by simple physical constraints. Despite this environment's simplicity compared to ours, its teachings about reward functions are salient.

Consider a reward which is inversely proportional to the distance from the goal. Using this reward, agents are incentivised to greedily minimize their distance from the goal. This does not work, however, because momentum needs to be built by first gathering potential energy in the opposite direction.

OpenAI defines the reward to be -0.1 for every step which does not reach the goal and $+100$ for the step which does reach the goal, which is well aligned with the overarching goal, but astoundingly sparse. Despite this sparseness, agents perform much better.
For our benchmark, a reward function which worked in practice was to reward the agent $+2$ for each step that it improves its last greatest height by at least a small value, and to penalize the agent -0.1 for each step that it did not do so. The advantages of this are that:

- Rewards are not proportional to the height, but still aligned with the notion that higher is better in the long term.
- The agent can justify moving indirectly towards the goal for a number of steps as long as this results in progress thereafter. The incentive balances the deterrent in this case.

We also added an especially high reward for reaching the top - y-coordinate above 36. We end the episode when either 1000 steps pass, a red object is hit, or the agent reaches the top

5.7 Benchmark Testing

We compared the performance of agents by their average rewards per step (higher is better). All data were gathered using 40 variants and 1000 iterations per template in training and testing. We used different reward functions in training but the same reward function in testing and evaluation.
We tested our benchmark quantitatively by performing experiments using hard-coded and reinforcement learning agents.

5.7.1 Hard-coded Agents

We had five hard-coded agents. Three take random actions, and two take actions according to carefully implemented heuristic scores.

Random Agents:
The random agent takes actions according to a given discrete probability distribution. An agent which takes actions purely at random would constantly contradict its own movements and was of no use to our analysis. We generally wanted our agents to go upwards, with some slight perturbations. Our random agents, named "leftward", "upward", and "rightward" take actions according to the following distributions:

Actions:	**LEFT**	**LEFT-UP**	**UP**	**RIGHT-UP**	**RIGHT**
Leftward	20%	50%	20%	10%	0%
Upward	10%	20%	40%	20%	10%
Rightward	0%	10%	20%	50%	20%

Heuristic Agents:
Heuristic agents take raycast observations as inputs and evaluate each action based on hard-coded rules. It then performs the best option indicated by scores.

Avoid Agent: The agent scores each option based on distances from itself to danger objects. It then chooses an action that maximizes the total distance from itself to danger objects.
Push Agent: The agent scores each option based on distances from itself to safe objects, or by prioritizing moving forward. It will choose an action that minimizes the total distance from itself to safe objects, or minimize its distance to the top when there are no nearby safe objects.

5.7.2 Reinforcement learning agents:

We tested the integration of our benchmark with Tensorflow-Agents by running a DQN and CDQN reinforcement learning agent on some of our tasks. We tried this using our different reward and observation approaches. Due to the time-dependent nature of this research direction, we were not able to be comprehensive in searching for optimal hyperparameters or network architectures. We list some that we considered and experimented with.

- Discount factor (gamma): The discount factor is a coefficient in the Bellman Equation. In theory, a value close to 0 causes the agent to prioritise immediate rewards, whereas a value close to 1 causes the agent to prioritise future rewards.

  $$Q(S_t, A_t) = (1 - \alpha)Q(S_t, A_t) + \alpha(R_t + \gamma \cdot \max_a(Q(S_{t+1}, a)))$$

  We chose a high value for gamma, to encourage the agent to form long term goals.

- Exploration factor (epsilon): We use a decaying epsilon-greedy strategy for agent-exploration.

- Reward design: In practice, the different rewards we had thought of had considerable impacts on the agent's learning. One surprise was that associating a high negative reward with hitting a red ball prevented agents from learning to avoid on the roof template.

6 Distribution of work (Q5):

During the development of our benchmark we organised weekly meetings to discuss the project. We did this to keep the group informed about progress, provide an opportunity to efficiently discuss ideas and opinions, and to delegate tasks which were suited to the strengths and interests of each individual. We aimed to let eachother know when we were going ahead with a particular contribution. Each individual found themselves able to work on the project at different times. If one individual was not able to contribute at a particular time, then the others would continue with development in their stead.

Our group was not without its faults when it came to teamwork. In future, we would consider using a more rigid development framework. We could have made more use of gitlab's 'issue' functionality, for example.

We list the individual contributions and ideas which we each take responsibility for:

6.1 Lachlan Stewart

- Early benchmark concept
- Created framework using Tensorflow, Box2D and Pygame for backend
- Formulated the abstract templates as reinforcement learning environments to interface with Tensorflow
- 'Variant' value concept and implementation
- Template designs
- Implemented Deep-Q and Categorical Deep-Q agents using Tensorflow
- Implemented Distribution agent
- Abstracted the observation in order to include multiple approaches
- Raycasting observation concept and implementation
- New-best-height reward concept and implementation
- Carried out experiments

6.2 Sihan Lin

- Template designs
- Benchmark framework design & refactoring
- Abstraction of agent & formalize agent which accept human input
- Structure agent action space & movement unit
- Abstract reward function and implement basic sparse reward functions
- Parameterized application launch arguments
- Implement efficient image observation & create DQN agent from it
- Conduct experiment on image-based DQN agent

6.3 Xin Lu

- Proposed action tasks and early benchmark concept
- Proposed idea of interactive physical 'puzzle'
- Proposed idea of agents conducting 'pushing' actions
- Template designs
- Implemented heuristic agent (avoidance)
- Implemented heuristic agent (pushing)
- Implemented image renderer and GIF convertor to visualize agent task performance
- Conducted experiments of heuristic agents and created quantitative graphs

7 Quantitative Experiment Results (Q6)

7.1 Hard-coded agents:

We performed some preliminary quantitative experiments to see whether simple strategies could solve benchmark tasks. We found that no strategy could solve all tasks, but some could perform well on selected templates. Appendix I and II include more details of templates and hard-coded agents' performances.

We focus first on the performance of the avoid agent on tasks which require the agent to avoid falling red objects. We see that in some templates the agent consistently outperforms the others, with high reward values ≈ 2 indicating that the agent reaches the top. In some other templates however, the agent does not perform consistently across all variants, getting stuck on static walls or being hit by a red ball.

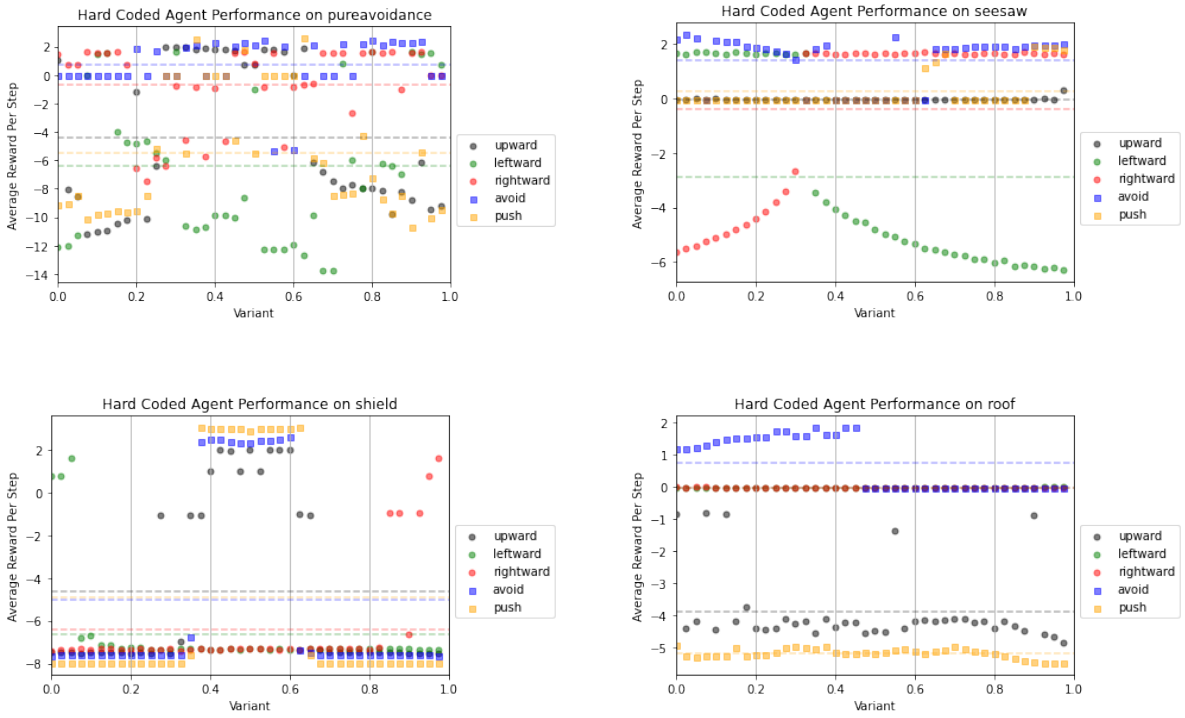


Figure 4: Performance comparison of non-intelligent agents in template PureAvoidance, Seesaw, Shield and Roof

Focusing on the push-agent, and templates[1] which require the agent to push blue objects, we see a similar mix of promising results and inconsistency.

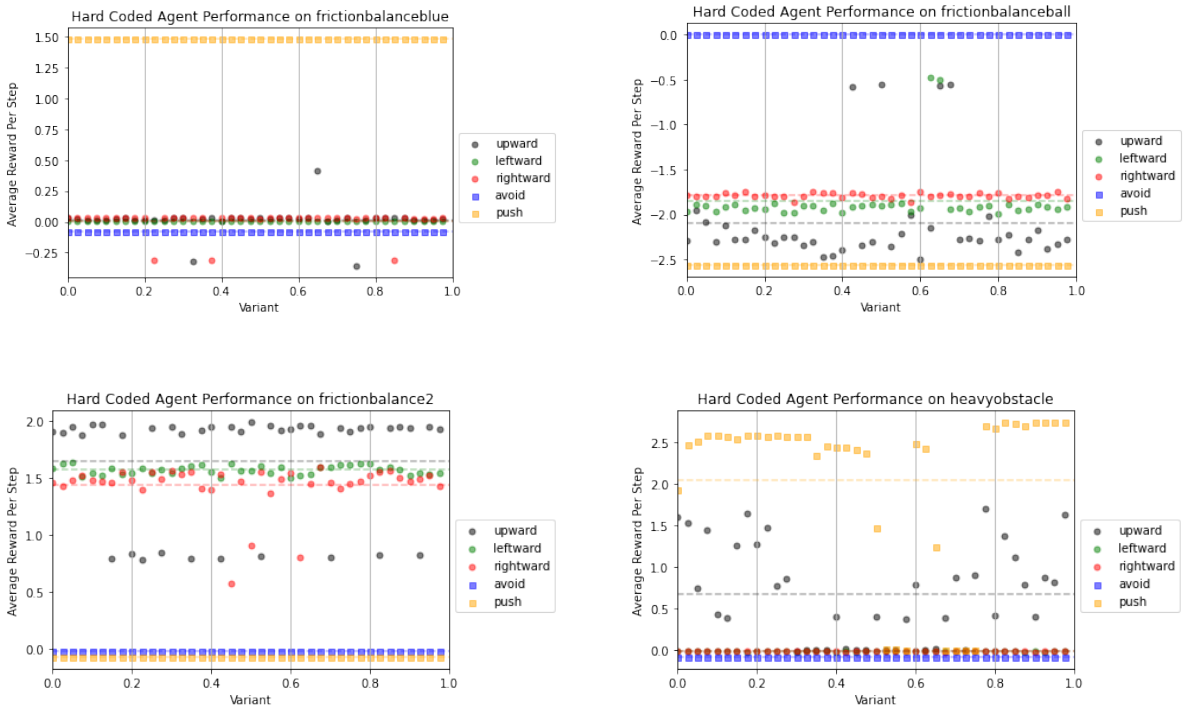


Figure 6: Performance comparison of non-intelligent agents in template FrictionBalanceBlue, FrictionBalanceBall, FrictionBalance2 and HeavyObstacle

[1]Due to the delicate nature of the constructions in the frictionbalance templates, the variant slider was not implemented for them, hence the deterministic agents perform exactly the same moving across the x-axis. We will change this in the future. we keep the chart design for visual consistency

Our analysis provides some key insights into our overall benchmark design.
Cross template:

- Through the testing of a range of different agents, we see that it is not simple to design an agent that can perform well across all templates (Figure 7). This is evidence that our benchmark as a whole is complex.
- Within a set of templates that demand a particular kind of behaviour, we see agents that perform well on some tasks but not well on others. This is evidence that our benchmark is testing different kinds of behaviour thoroughly.

Within template:

- For some templates, some variants are significantly easier than others. We see this in the shield template, where $0.4 < v < 0.6$ sees some agents perform well.
- The observed performance differences with respect to variants show that our design of variants challenges agents that do not understand the environment's physical dynamics to perform well in all variants.
- An analysis like ours helps identify bias in the template design towards particular strategies, such as going left vs right. We see this in the Seesaw template which slightly favours going right. One should be careful to use the variant to reduce bias.

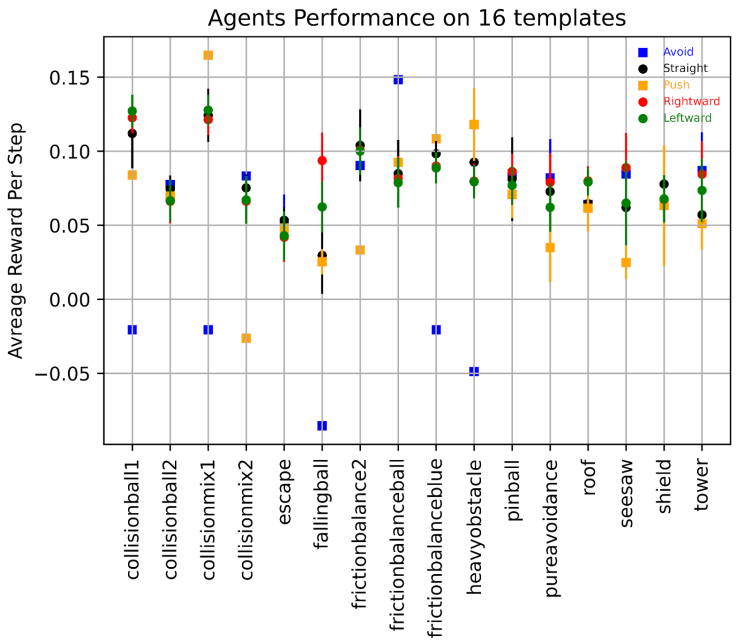


Figure 7: Average Reward Per Step (ARS) of agents across all current benchmark templates. Vertical bars show the standard deviation of ARS from all variants. Total reward after episode ends is set to zero to fairly compare different episode lengths.

7.2 Deep learning agent:

We trained a CDQN for 60,000 timesteps on the Seesaw template and managed to marginally improve on the best hardcoded agent performance across most templates. There is much still to be done in this area, but we are glad to demonstrate that reinforcement learning *can* improve on hard-coded strategies. In some test runs of this CDQN, the agent demonstrates patience beneath the seesaw and waits for objects to fall.

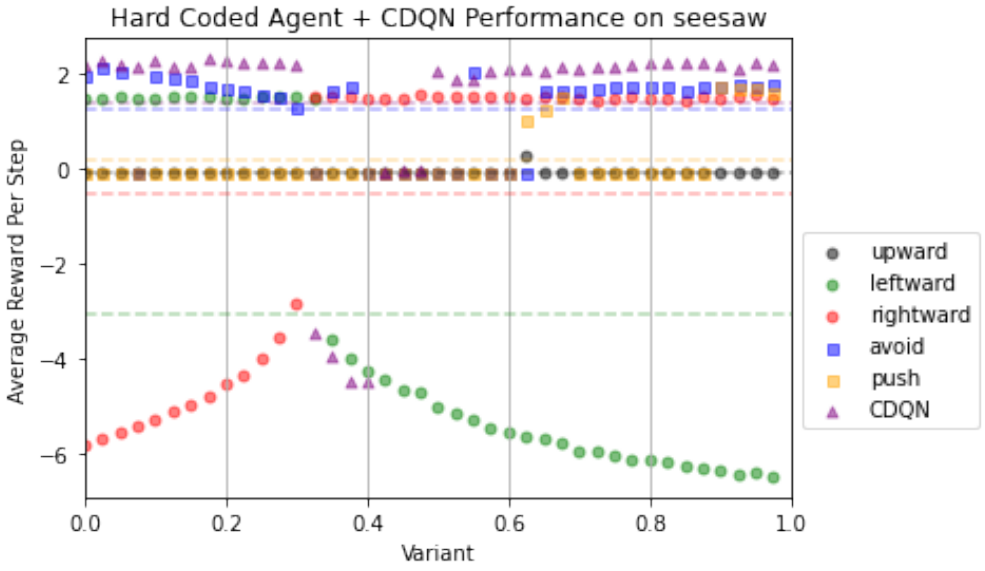


Figure 8: Performance of the reinforcement learning agent (pink triangles) on the Seesaw template compared to the non-intelligent agents

8 Future research (Q7):

We identify several ways our benchmark can be extended:

8.1 Continuous Action Space:

On our benchmark, the action space of the agent is discrete. The agent can move in 8 directions at a fixed velocity. Some benchmarks in reinforcement learning allow the agent to take actions from a continuous space, and ours could be easily adjusted for this by allowing the velocity to exist on a closed interval ending at the original fixed velocity.

8.2 Collecting Human Data:

We would like to add a feature to our benchmark tools which facilitates the collection of human experience. We propose that a dataset of (observation, action, reward) triples collected through human play would provide:

- A training dataset for performing supervised learning in RL models.
- A baseline of our benchmark to serve as a medium to long term goal for future research.
- An indication of template difficulty.

8.3 Improved Templates:

We would like to conduct further experiment on our benchmark by:

- Using more advanced reinforcement learning models to solve our benchmark.
- Training the models for more iterations with better computing resources.
- Using a more efficient strategy for environment exploration.

8.4 Alternate Deep Q Network architectures:

8.4.1 Convolutional-Neural-Network (CNN):

Convolutional-Neural-Network architectures are standard in computer vision research and applications. We believe that an attempt should be made to use a CNN for the image-input observation approach.

8.4.2 Long short-term memory (LSTM):

We believe that a recurrent network architecture could imbue the agent with the ability to plan ahead. LSTM units have been performing highly in recent research. An agent with recurrent units could theoretically develop the capacity to identify and remember important events from past states, and extrapolate the motion of objects without needing multi-frame input. This could reasonably be applied to either of our observation approaches.

8.4.3 Variational-Auto-Encoders (VAE):

Variational-Auto-Encoders are a network architecture used for generative models. VAEs compress inputs to a lower-dimensional space. We suggest that the encoder could be used for identifying key physical information in the observational input, and that this aspect of learning could be systematically separated from the reinforcement learning process. For example, human play could generate an observational dataset to train a VAE, and the encoder half of the VAE could be used as the front-end to a deep-Q network.

9 Reflection (Q8):

9.1 Technical Skills

In the process of producing our own benchmark to address our research question, we have studied the various technologies and framework and adapted them into our project, such as learning how to use Box2D to construct our environment, and how to use TensorFlow to create agent implementation to experiment our benchmark. Moreover, we have studied a range of state-of-the-art research for solving intuitive physics problems, while gaining a more in depth understanding of the reinforcement learning algorithm. Particularly, we have conducted our own experiment to test the performance impact of different observations and reward function designs, which is a valuable experience to use to conduct further research in the topic of reinforcement learning.

9.2 Research Skills

Our team also learned many non-technique experiences in the process of conducting professional research as a team. Firstly, by organizing regular meetings to discuss our research topics, we have learned how to organize professional research discussion. Furthermore, we have also learned to conduct individual literature reviews to find relevant research in the topic, which is an important skill for performing real research. We also developed critical thinking skills in the project by analyzing the pros and cons of our own idea or other member's idea, which greatly enriched the diversity of our research (such as introducing different types of observation and reward functions). Most importantly, we have studied how to conduct professional quantitative experiments to validate our benchmark design, and generate conclusions based on analysis of the experiment outcome.

9.3 Future improvements

On the other hand, there are a range of things we did not perform well in our teamwork, which could be further improved. Firstly, our team lacks a team management process. Because there is no clear leadership in our team, we are mainly motivated by personal interests and no hard deadline for each individual task. As a result, the team does not have consistent working velocity and the clear task distribution between different members. In the future, we should set up clear responsibility roles (such as team leader, developer...) for each member in the team, so that it will be far more efficient to manage and cooperate with the tasks of different members.

Further, our communication process lacks transparency, which raises the conflicts and misunderstanding between team members. For example, when the progress of tasks is not well-informed to other team members, it is difficult for other members to keep track of uncompleted tasks. More importantly, it may result in two team members working on the same task and wasting an individual's time. In future teamwork, we should clearly document every task progress from the different members, so that the teamwork can be synchronized between members.

10 References

Bakhtin, A., van der Maaten, L., Johnson, J., Gustafson, L. and Girshick, R., 2019. Phyre: A new benchmark for physical reasoning. *Advances in Neural Information Processing Systems, 32.*

Duan, J., Dasgupta, A., Fischer, J. and Tan, C., 2022. A Survey on Machine Learning Approaches for Modeling Intuitive Physics. *arXiv preprint arXiv:2202.06481.*

Januszewski, P. (2021) 'Best Benchmarks for Reinforcement Learning: The Ultimate List', Neptune, 15 December 2021 [Blog]. Available at `https://neptune.ai/blog/best-benchmarks-for-reinforcement-learning` (Accessed 30 May 2022)

Tassa, Y., Doron, Y., Muldal, A., Erez, T., Li, Y., Casas, D.D.L., Budden, D., Abdolmaleki, A., Merel, J., Lefrancq, A. and Lillicrap, T., 2018. Deepmind control suite. *arXiv preprint arXiv:1801.00690.*

Xue, C., Pinto, V., Gamage, C., Nikonova, E., Zhang, P. and Renz, J., 2021. Phy-q: A benchmark for physical reasoning. arXiv preprint arXiv:2108.13696.

Peterson J, *Solving the Problem of Perception: Jordan Peterson at Cambridge* [Video], YouTube, `https://www.youtube.com/watch?v=5HgSnS-z4JU`

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., ... & Hassabis, D. (2015). *Human-level control through deep reinforcement learning.* nature, 518(7540), 529-533.

Moore. A, 1990, Efficient Memory-based Learning for Robot Control, University of Cambridge, `https://www.cl.cam.ac.uk/techreports/UCAM-CL-TR-209.pdf`

Stapelberg, B., & Malan, K. M. (2020). A survey of benchmarking frameworks for reinforcement learning. *South African Computer Journal*, 32(2), 258-292.

11 Appendix I - Templates

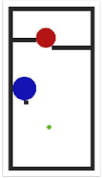
11.1 Table

Template Name	Primary Reasoning Required	Strategy	Description
Seesaw	Mass	Choose correct path	A blue beam balances two red objects of differing mass. One path will be blocked by the falling objects
Heavy Obstacle	Mass	Push blues / choose correct path	Two boxes of differing mass need to be pushed. One is lighter, which makes it easier to move.
Roof	Structure	Patience / avoid reds	A teepee-like blue structure encloses the agent. Two balls fall from a small opening at the top.
Tower	Structure	Push blues / avoid reds	A tower built of red and blue objects encloses the agent. The agent must dismantle it safely.
Friction Balance Blue, Friction Balance 2	Friction, Structure	Push blues / unbalance the friction structure	A friction and structure puzzle. The friction structure is blocking the agent's way to the top and must be temporally unbalanced by pushing the blues. There is more than one way to unbalance the structure. However, unbalancing from the right will be the fastest.
Friction Balance Ball	Friction, Structure	Push blues / avoid reds	A friction and structure puzzle. The friction structure is blocking the agent's way to the top and must be temporally unbalanced by pushing the blues. Also the agent need to carefully avoid the falling reds. There is more than one way to unbalance the structure. However, unbalancing from the right will be the fastest and safest.
Collision Ball 1	Utilization	Push blues / avoid reds / clear obstacles	A utilization and obstacle puzzle. Agent should push the blue to push away the blocking red ball. It also needs to wait until it can safely move forward without colliding with the red.
Collision Ball 2	Utilization	Push blues / avoid reds / clear obstacles	A utilization and obstacle puzzle. Agent should push the blue to push away the blocking red balls. There is more than one way to clear the obstacles. However, clearing from the right will be the safest.

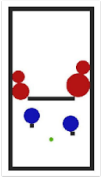
Collision Mix 1, Collision Mix 2	Utilization	Push blues / avoid reds / clear obstacles	A utilization and obstacle puzzle. Agent should push the blue to push away the blocking red balls. There is more than one way to clear the obstacles, and each choice has no obvious advantages.
Pure Avoidance	Collisions/ trajectories	Hide behind blues and statics / avoid reds	A multitude of different blue and red shapes fall in a chaotic fashion. The agent must navigate them.
Pinball	Trajectories	Predict trajectory of falling balls	Multiple red balls are falling to a pinball playground. The agent must perform strong reasoning about the future trajectory of balls to avoid all balls.
FallingBall	Collisions	Move left if the blue ball will not collide with red ball with enough size	A blue ball (size is vary) is falling from the top and touching a red ball which originally blocks the right path; if the blue ball is large enough, the red ball will be pushed left and block the left path.
Shield	Utilization	Hide behind blues	6 red balls above a horizontal blue beam fall. The beam can shelter the agent.
Escape	Utilization	Push blues	The agent is enclosed by a static roof and two red walls. It must use a small blue object to shift the walls and escape.

11.2 Images

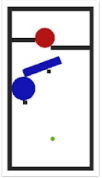
Collisionball1



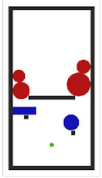
Collisionball2



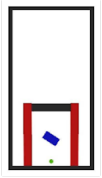
Collisionmix1



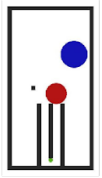
Collisionmix2



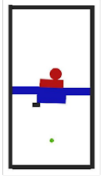
Escape



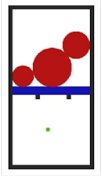
Fallingball



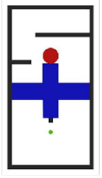
Frictionbalance2



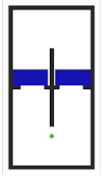
Frictionbalanceball



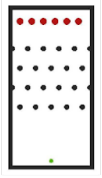
Frictionbalanceblue



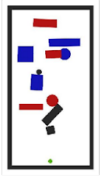
Heavyobstacle



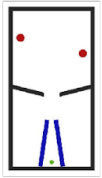
Pinball



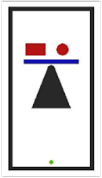
Pureavoidance



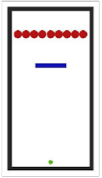
Roof



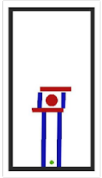
Seesaw



Shield



Tower



12 Appendix II - Baseline Non-Intelligent Agents Performance on All Templates

We compared the performance of agents by their average rewards per step (higher is better). All data were gathered using 40 variants and 1000 iterations per template in training and testing. We used different reward functions in the training process but the same reward function in testing and evaluation. Standard deviation shows how performance differs around all variants.