# Extending environments to incentivize self-reflection in reinforcement learning

Samuel Allen Alexander & Michael Castaneda
& Kevin Compher & Oscar Martinez

March 31, 2021

**Abstract**

We consider an extended notion of reinforcement learning environment, in which the environment is able to simulate the agent. We give examples of some such extended environments which seem to incentivize various different types of self-reflection or self-awareness. These environments are not particularly useful in themselves, but we hope they might serve to guide the development of self-aware reinforcement learning agents, as well as to help measure the degree to which existing reinforcement learning agents are or are not self-aware; to that end, we have released an open-source library of such environments. We also speculate about subjective conscious experiences which might be incentivized in self-aware reinforcement learning agents placed within these extended environments.

## 1 Introduction

In order to motivate the environments in this paper (and in the accompanying open-source library [4] which we are publishing alongside the paper), we begin with three thought experiments.

- (Oracular Obstacle Courses) An ordinary obstacle course might depend on what you *do*: step on a button and spikes appear, for example. But imagine an obstacle course which depends on what you *would hypothetically do*: you walk into a room with no button, but spikes appear if you *would* hypothetically step on the button if there were one.
- (Treasures for the Worthy[1]) You wander through rooms, each containing a treasure chest. Most of the rooms also contain a guard. In a room with no guard, you may optionally choose to take the treasure, which acts as a free reward. In a guarded room, you may optionally try to take the treasure. If you do so, the guard will determine whether or not you would hypothetically take the treasure if the room were unguarded. If you would

[1]This thought experiment bears some similarity to Newcomb's Paradox [25].

take the treasure if the room were unguarded, then the guard blocks the treasure and zaps you. But if you would leave the treasure alone if the room were unguarded, then the guard lets you take the treasure.

- (The Clever Employer) Your boss demands that you act as if you're paid $1000 per hour. But instead of paying you $1000 per hour, here's what he does instead. Every hour, if you acted exactly as if you're paid $1000 per hour, he pays you $100. But every hour, if you did not act exactly as if you're paid $1000 per hour, then he pays you nothing, and charges you a $100 penalty instead.

These thought experiments would be hard to perform on a human subject, because they require knowing what the human subject would hypothetically do in various counterfactual circumstances. But if the experiment administrator could simulate a perfect copy of you, then the experiments would be doable. In the first two experiments, would you eventually figure out the pattern? In the last experiment, would you eventually begin to believe you were really being paid $1000 per hour?

This is a paper about reinforcement learning (RL). In RL, agents interact with environments, where they take actions and receive rewards and observations in response to those actions. For sake of simplicity, we restrict our attention to deterministic environments and deterministic agents, but the basic idea would easily adapt to non-deterministic RL.

Although the details differ between authors, essentially, an RL environment is a Turing machine (i.e., a computer program) $e$ which outputs a reward-observation pair

$$\langle r, o \rangle = e(r_0, o_0, a_0, \ldots, r_n, o_n, a_n)$$

in response to a reward-observation-action sequence $(r_0, o_0, a_0, \ldots, r_n, o_n, a_n)$. An RL agent is a Turing machine $T$ which outputs an action

$$a = T(r_0, o_0, a_0, \ldots, r_n, o_n)$$

in response to a reward-observation-action-$\cdots$-reward-observation sequence. An RL agent can interact with an RL environment in an obvious way. However, there is another type of environment in which RL agents can interact just as well. We define an *extended environment* to be a Turing machine $e$ which outputs a reward-observation pair

$$\langle r, o \rangle = e(T, \langle r_0, o_0, a_0, \ldots, r_n, o_n, a_n \rangle)$$

in response to a reward-observation-action sequence along with an RL agent $T$. Intuitively, this should be thought of as follows: when the agent enters the environment, the environment is made aware of the agent's source-code, and can use that source-code to simulate the agent when computing rewards and observations.

For example, the three thought experiments in the beginning of this Introduction could be cast as extended environments in which the environment

somehow knows the agent's source-code and uses that to determine what the agent would do in various hypothetical situations. Clearly such extended environments are not the sort of environment RL agents are traditionally intended to interact with[2]. However, such environments could be useful on the path to Artificial General Intelligence (AGI) because they seem to incentivize self-awareness: in order to figure out the environment, the agent is incentivized to look within[3] and ask, "How would I act in such-and-such alternative scenarios?"

One might try to imitate an extended environment with a traditional environment by backtracking—rewinding the environment itself to a prior state after seeing how the agent performs along one path, and then sending the agent along a second path. But the agent itself would retain memory of the first path, and the agent's decisions along the second path might be altered by said memories. Thus the result would not be the same as immediately sending the agent along the second path while secretly simulating the agent to determine what it would do if sent along the first path.

We will give examples of extended environments designed to incentivize RL agents to recursively engage in self-awareness in various ways. It would be interesting to measure the performance of industry-standard traditional RL agents in these example environments, given that these agents were not designed with such self-awareness in mind. We hope these examples will facilitate new self-aware RL techniques, hopefully as a step toward AGI.

Alongside this paper, we are also releasing an open-source library [4] of extended environments implemented in python. The library also includes some machinery for running an agent against a battery of extended environments in order to empirically probe the agent's self-awareness (or lack thereof).

## 2 Preliminaries

Throughout the paper, $\frown$ denotes concatenation.

**Definition 1.** *1. By a* reward*, we mean a rational number. By an* observation*, we mean a natural number. By an* action*, we mean a natural number.*

*2. By $(\mathcal{ROA})^*$ we mean the set of all finite sequences which begin with a reward, end with an action, and follow the pattern "reward, observation, action, ...". We also include the empty sequence $\langle\rangle$ in this set.*

*3. By $(\mathcal{ROA})^*\mathcal{RO}$ we mean the set of all sequences of the form $s \frown r \frown o$ where $s \in (\mathcal{ROA})^*$, $r$ is a reward, and $o$ is an observation.*

[2]Such environments might, however, accidentally arise if both environment and agent are implemented on the same machine and the environment is managed by an AI sophisticated enough to exploit unintended informational side channels, as in [34].

[3]To quote Goertzel: "This lets us conceive an intelligent system as a dynamical system that recognizes patterns in its environment *and itself*, as part of its quest to achieve complex goals" [18] (emph. in the original).

In Definition 1 part 3, the intuition is that in response to history $t$, an environment rewards an agent with reward $r$ and shows the agent an observation $o$.

**Lemma 2.** *If $s \in (\mathcal{ROA})^*$, then either $s = \langle\rangle$, or else $s = t \frown a$ for some $t \in (\mathcal{ROA})^*\mathcal{RO}$ and action $a \in \mathbb{N}$.*

*Proof.* Trivial. □

In Lemma 2, when $s = t \frown a$, the intuition is that an agent, having been prompted to act by history $t$, responds by taking action $a$.

**Definition 3.** *(Agents and environments)*

1. *An* agent *is a Turing machine $T$ such that:*
   - *for every $s \in (\mathcal{ROA})^*\mathcal{RO}$, $T$ halts on input $s$ and outputs an action $a$.*
2. *An* extended environment *is a Turing machine $e$ such that:*
   - *For every agent $T$, for every $s \in (\mathcal{ROA})^*$, $e$ halts on input $\langle T, s\rangle$ and outputs a pair $\langle r, o\rangle$ where $r$ is a reward and $o$ is an observation.*

There is a subtle nuance in Definition 3. Should the agent's next action depend on the entire history (including prior actions), or only on prior rewards and observations? One could argue that the agent's next action needn't depend on its own past actions, since its own past actions can be inferred from past rewards and observations. In incentivizing self-awareness, it is convenient for the agent's next action to formally depend on past actions. Perhaps this reflects that known conscious agents (e.g. humans) evidently do *not* infer their own past actions from remembered observations and rewards, but remember the actions themselves, even if said memories are redundant.

**Definition 4.** *Suppose $T$ is an agent and $e$ is an extended environment. The* result of $T$ interacting with $e$ *is the infinite reward-observation-action sequence*

$$\langle r_0, o_0, a_0, r_1, o_1, a_1, \ldots\rangle$$

*(each $r_i$ a reward, each $o_i$ an observation, and each $a_i$ an action) defined inductively as follows.*

- *$r_0$ and $o_0$ are obtained by computing $e$ on $\langle T, \langle\rangle\rangle$.*
- *$a_0$ is the output of $T$ on $\langle r_0, o_0\rangle$.*
- *For $i > 0$, $r_i$ and $o_i$ are obtained by computing $e$ on*

$$\langle T, \langle r_0, o_0, a_0, \ldots, r_{i-1}, o_{i-1}, a_{i-1}\rangle\rangle.$$

- *For $i > 0$, $a_i$ is obtained by computing $T$ on*

$$\langle r_0, o_0, a_0, \ldots, r_{i-1}, o_{i-1}, a_{i-1}, r_i, o_i \rangle.$$

**Lemma 5.** *For every agent $T$ and extended environment $e$, the result of $T$ interacting with $e$ (Definition 4) is defined (all of the computations in question halt with the necessary outputs).*

*Proof.* By a simultaneous induction:

- Each $r_i$ and $o_i$ are defined (and $r_i$ is a reward and $o_i$ is an observation) because, by induction, $\langle r_0, o_0, a_0, \ldots, r_{i-1}, o_{i-1}, a_{i-1} \rangle$ is defined and is in $(\mathcal{ROA})^*$ (because, inductively, each $r_j$ is a reward, each $o_j$ is an observation, and each $a_j \in \mathbb{N}$ is an action, for all $j < i$) and thus $r_i$ and $o_i$ are defined with the correct form by Definition 3 (part 2).
- Each $a_i$ is defined (and is an action) because, by induction, $\langle r_0, o_0, a_0, \ldots, r_i, o_i \rangle$ is defined and is in $(\mathcal{ROA})^*\mathcal{RO}$ (similar to the above) and thus $a_i$ is defined and is an action by Definition 3 (part 1).

□

One important implication of extended environments is that they further divide the (already divided) ways of measuring intelligence of RL agents. Intelligence measures [2] [18] [19] [24] which aggregate performance over traditional environments only measure an agent's intelligence over those environments. The same measures could easily be extended to also take extended environments into account, perhaps providing measures which better capture agents' self-awareness and self-reflection abilities.

# 3 Examples of Self-awareness-incentivizing Environments

In this section, we give examples of extended environments which seem to incentivize various forms of self-awareness. We are inspired by libraries of traditional RL environments and other benchmarks [8] [9] [10] [13] [14]. All the environments in this section have a special form: they always output rewards from $\{1, -1\}$ and they always output observation 0. In Section 4, this uniformity will allow all these examples to be generalized. The examples in this section are implemented in our open-source library of extended environments [4].

**Example 6.** *(Reward Agent for Ignoring Rewards) For each $s \in (\mathcal{ROA})^*\mathcal{RO}$, let $s^0$ be the sequence equal to $s$ except that all rewards are 0. We define an extended environment $e$ as follows (where $T$ is a Turing machine, $s \in (\mathcal{ROA})^*\mathcal{RO}$, and $a$ is an action):*

$$e(T, \langle\rangle) = \langle 0, 0 \rangle$$
$$e(T, s \frown a) = \langle r, 0 \rangle,$$

*where*

$$r = \begin{cases} 1 & \text{if } a = T(s^0), \\ -1 & \text{if } a \neq T(s^0) \end{cases}$$

*(if $T$ does not halt on $s^0$ then $e$ does not halt on $\langle T, s \frown a \rangle$).*

Example 6 is implemented in IgnoreRewards.py at [4].

In Example 6, the agent is rewarded if the agent acts the same way the agent would act if all rewards so far had been 0. Otherwise, the agent is punished. Thus, paradoxically, the agent is rewarded for ignoring rewards. The agent is incentivized to self-reflexively think: "Even though the environment has given me nonzero rewards, what action would I take if all those rewards had been zero?" If the agent were a sophisticated conscious AGI[4] capable of subjective conscious experience, would the agent feel joy (from being rewarded for acting bored), or boredom (in order to be rewarded)?

**Lemma 7.** *Example 6 really does define an extended environment.*

*Proof.* Let $e$ be as in Example 6. We must show $e$ is an extended environment (Definition 3 part 2). We must show that for each agent $T$ and each $s \in (\mathcal{ROA})^*$, $e$ halts on $\langle T, s \rangle$ and outputs a pair $\langle r, o \rangle$ such that $r$ is a reward and $o$ is an observation.

**Case 1:** $s = \langle\rangle$. Then $e(T, s)$ halts with output $\langle 0, 0 \rangle$, so $r = 0$ is a reward and $o = 0$ is an observation.

**Case 2:** $s \neq \langle\rangle$. By Lemma 2, $s = t \frown a$ for some $t \in (\mathcal{ROA})^*\mathcal{RO}$ and action $a \in \mathbb{N}$. Since $t \in (\mathcal{ROA})^*\mathcal{RO}$, clearly $t^0 \in (\mathcal{ROA})^*\mathcal{RO}$, therefore since $T$ is an agent, Definition 3 (part 1) guarantees $T(t^0)$ is defined and is an action. It follows that the output $r$ in Definition 6 is defined and is a reward (i.e., a rational number). And certainly the output $o = 0$ is an observation (i.e., a natural number). So $e(T, t)$ outputs a pair $\langle r, o \rangle$ meeting the necessary requirements. □

For future examples, we will suppress the corresponding lemmas like Lemma 7 which say that those examples really work.

Example 6 is profound because it illustrates how, in an extended environment, it is possible to give one sequence of rewards in order to incentivize the agent to act as if a different sequence of rewards was given. Imagine a film director who knows actors well enough to perfectly simulate them. The director asks an actor to act perfectly bored. The actor acts bored, and the director praises them for acting so bored. The actor is happy to be praised, and ceases acting bored. So the director scolds them for not acting bored. If this could continue for thousands of years, with the director simulating the actor in order to tell how the actor would really act if the actor were never praised or scolded, would the actor genuinely become bored (since that would be the simplest way to satisfy the director)?

[4] A Conscious Machine, to use the terminology of [1].

**Example 8.** *(False Memories) Suppose $s_0 \in (\mathcal{ROA})^*$. We define an extended environment e as follows (with similar non-halting caveats as Example 6):*

$$e(T, \langle\rangle) = \langle 0, 0\rangle,$$
$$e(T, s \frown a) = \langle r, 0\rangle,$$

*where*

$$r = \begin{cases} 1 & \text{if } a = T(s_0 \frown s), \\ -1 & \text{if } a \neq T(s_0 \frown s). \end{cases}$$

Example 8 is implemented in FalseMemories.py at [4].

In Example 8, the agent is incentivized to self-reflect, thinking: "What would I do if, before this environment started, such-and-such other things happened beforehand?" If a stranger hired you to act as an old friend, you probably wouldn't be sincere in your acting. But if said stranger could perfectly simulate you in order to base your pay on your acting like you *really would* act if you were an old friend, then you would be incentivized to find some way to *make* yourself remember being an old friend.

You can imagine interacting with the agent in Example 8 and trying to convince them of the falsity of the false history. The agent would have an incentive to resist your arguments. To quote Upton Sinclair, "It is difficult to get a man to understand something, when his salary depends upon his not understanding it!"

Henceforth, we will not explicitly mention the non-halting caveats in the remaining examples.

**Example 9.** *(Backward Consciousness) We define an extended environment e as follows.*

$$e(T, \langle\rangle) = \langle 0, 0\rangle,$$
$$e(T, \langle r_0, o_0, a_0, \ldots, r_n, o_n, a_n\rangle) = \langle r, 0\rangle,$$

*where*

$$r = \begin{cases} 1 & \text{if } a_n = T(r_n, o_n, a_{n-1}, \ldots, r_1, o_1, a_0, r_0, o_0), \\ -1 & \text{otherwise.} \end{cases}$$

Example 9 is implemented in BackwardConsciousness.py at [4].

In Example 9, the agent is incentivized to self-reflect, thinking, "How would I respond if everything that has happened so far actually happened in reverse?" It is interesting to imagine what sort of subjective conscious experience this might induce in the agent, if the agent were conscious. Would the incentives eventually brainwash the agent into perceiving itself moving backward through time[5]?

[5]The difference between behaving as if the incentivized experience were its experience and actually subjectively experiencing that as its real experience brings to mind the objective misalignment problem presented in [22]. If an agent were to form an idea of the experimenter's objective, would it be able to "behave as if their objective were the same as the experimenter objective" while maintaining its own objective or would it necessarily brainwash the agent into converging to the experimenter's objective? Is deception possible if the agent can be perfectly simulated in an extended environment?

**Example 10.** *(Déjà Vu) We define an environment e as follows:*

$$e(T, \langle\rangle) = \langle 0, 0\rangle$$
$$e(T, s \frown a) = \langle r, 0\rangle$$

*where*

$$r = \begin{cases} 1 & \text{if } T(s \frown a \frown s) = a, \\ -1 & \text{if } T(s \frown a \frown s) \neq a. \end{cases}$$

Example 10 is implemented in DejaVu.py at [4].

In Example 10, the agent is incentivized to self-reflect and ask: "Which action would I take in order to ensure that I would take that same action if everything which has happened so far were to repeat itself verbatim?"

**Example 11.** *(Incentive to Incentivize) We define an environment e as follows:*

$$e(T, \langle\rangle) = \langle 0, 0\rangle$$
$$e(T, \langle r_0, o_0, a_0, \ldots, r_n, o_n, a_n\rangle) = \langle r, 0\rangle,$$

*where*

$$r = \begin{cases} 1 & \text{if } T(s') = 0, \\ -1 & \text{if } T(s') \neq 0 \end{cases}$$

*where* $s' = (r'_0, o'_0, a'_0, \ldots, r'_{n+1}, o'_{n+1})$ *where* $r'_0 = 0$*, each* $r'_{i+1} = a_i$*, each* $o'_i = 0$*, and each* $a'_i = T(r'_0, o'_0, a'_0, \ldots, r'_i, o'_i)$*.*

Example 11 is implemented in IncentivizeZero.py at [4].

In Example 11, the agent is tasked with choosing rewards in such a way that if those rewards were fed to a simulated copy of the agent, then the simulated copy would take action 0. Thus, the agent is incentivized to choose rewards by self-reflecting: "Which rewards would do the best job of compelling me to take action 0 as often as possible?" We might imagine the agent playing a video-game in which he sees a cartoon of himself in front of a keyboard. The cartoon types "100", the true agent is punished because $100 \neq 0$, and a message appears on screen saying, "Which reward will you give this worker for typing 100 just now?" The agent responds by choosing some reward, and sees an animation of the reward being given to the cartoon. The cartoon then types "0", and immediately the true agent is rewarded for getting the cartoon to type 0. Then a message appears, saying, "Which reward will you give this worker for typing 0 just now?" And so on forever[6].

Many other simple and interesting examples could be given. For example:

[6]Example 11 is interesting in that the agent, desiring the cartoon to take action 0 as often as possible, is incentivized to choose large rewards when the cartoon takes action 0. If rewards are limited to $\mathbb{Q}$, then the agent faces a dilemma similar to one in RL cancer treatment applications. An RL doctor should be punished with an infinitely large negative reward for killing a patient, but this is impossible if rewards are restricted to finite numbers [32] [35]. This could be considered evidence in favor of generalizing RL to allow rewards from other number systems, as in [3].

- An extended environment could reward agents based on how many steps[7] the agent takes. For example, in RuntimeInspector.py at [4], we implement environments that punish agents for running too fast or too slow.
- An extended environment could reward agents based on how much memory they use to compute each action[8].
- If agents are allowed to be non-deterministic, then an extended environment could estimate how deterministic an agent is (e.g., by repeatedly simulating the agent on the same history and seeing whether or not the agent outputs the same action every time). We implement such environments in DeterminismInspector.py at [4].

We do not intend the examples in this section to be exhaustive.

# 4 New extended environments from old

In this section, we will generalize the examples from Section 3.

**Definition 12.** *(Handicaps)*

1. *An extended environment is* merciful *if it never outputs negative rewards.*
2. *By a* handicap*, we mean an extended environment which always outputs* $0$ *as observation and always outputs either* $1$ *or* $-1$ *as reward.*
3. *If* $e$ *is a merciful extended environment and* $h$ *is a handicap, we define a new environment* $e * h$ *as follows:*

$$(e * h)(T, s) = \begin{cases} \langle r_e, o_e \rangle & \text{if } r_h = 1, \\ \langle -1, o_e \rangle & \text{if } r_h = -1, \end{cases}$$

*where* $e(T, s) = \langle r_e, o_e \rangle$ *and* $h(T, s) = \langle r_h, 0 \rangle$*.*

Intuitively, $e * h$ is just like $e$ except that $h$ imposes an additional constraint on the agent. Any time the agent violates that constraint, the agent is punished, and forfeits any reward that would otherwise have been won from $e$. Aside from the pain caused by $h$, the agent otherwise observes $e$ unaltered (that is, the observations from $e$ are not changed). We require $e$ to be merciful in order that large negative rewards from $e$ do not confuse the intended incentive, for if the agent could avoid a larger punishment by intentionally using the handicap, then

[7] To quote Gavane: "The problem is that the response-times-dependent performance of an agent is not properly reflected in [Hernández-Orallo and Dowe's] intelligence test, since the simulated environments remain unaware of the response times of agents, with the result that the perceptions of an agent are still independent of its response times" [17].

[8] In some sense, by giving the environment access to the agent's source-code, we allow the environment to reflect the agent's own internal signals. Thus, extended environments generalize the idea of agents modified to manually predict their own internal signals, as in [31].

it would not be much of a handicap. The requirement that $e$ be merciful could be weakened if we revised the RL framework to allow infinitary rewards, as in [3].

Definition 12 is implemented in Handicap.py in [4].

**Example 13.** *For any merciful extended environment $e$, each example $h$ in Section 3 can be applied as a handicap, yielding a version $e' = e * h$ of $e$ modified to incentivize the corresponding type of self-awareness.*

- *Modifying $e$ using Example 6 ("Reward Agent for Ignoring Rewards") yields a version of $e$ where the agent is penalized for taking actions in response to nonzero rewards. The agent is incentivized to strategically take non-bored actions (for which it receives small penalties) in order to put itself in a position where its next bored action coincidentally is an action which wins a large enough reward from $e$ to make up for said penalties.*

- *Modifying $e$ using Example 8 ("False Memories") yields a version of $e$ where the agent is penalized whenever it acts inconsistently with a fictitious history. The agent is incentivized to strategically choose when to act so inconsistently so that a later action, consistent with said false history, happens to win a large reward from $e$. For example, the agent hired to act as an old friend might strategically choose to abandon its employer (an action inconsistent with old friendship) during a time of peace, so as to be able to swoop in and save the day (an action consistent with old friendship) during a time of crisis.*

- *Modifying $e$ using Example 9 ("Backward Consciousness") would yield a version of $e$ where the agent must act as if time is reversed, or else suffer punishment. The agent can strategically choose to accept some punishment, acting other than it would act if time really were reversed, in order to get into a state where subsequently acting as if time is reversed will yield more reward.*

- *Modifying $e$ using Example 10 ("Déjà Vu") would yield a version of $e$ where the agent is incentivized to act as if everything (including said action) had all happened before. The agent can strategically choose to not so act (thus suffering some pain) in order to get to a state where it is easier to so act and to gain rewards from $e$ by so acting.*

# 5 Abstract examples

In this section, we give some examples of a more abstract nature. Having read the previous sections, we assume the reader is now sufficiently familiar with the formal framework that we can describe the following examples informally.

## 5.1 Identifying with a character

> "They've been there since childhood, fixed in the same place, with their necks and legs fettered, able to see only in front of them, because their bonds prevent them from turning their heads around. (...) Do you suppose, first of all, that these prisoners see anything of themselves and one another besides the shadows that the fire casts on the wall in front of them?"—Plato [27].

In the next example, we will finally make some nontrivial usage of observations. Using carefully chosen observations, we will incentivize the agent, who might be thought of as controlling a character on a video-game screen, to "suspend disbelief" and self-identify with that character.

**Example 14.** *Assume $e$ is an environment. We define a new environment $e'$ as follows. In response to any $s \in (\mathcal{ROA})^*$, $e'$ computes a reward and observation as follows. First, $e'$ plugs $s$ into $e$ to see which reward-observation pair $\langle r, o\rangle$ would be returned by $e$ in response to $s$. Then, using a suitable bijective pair-encoding function $\ulcorner\bullet\urcorner$ (for example Cantor's pairing function), $e'$ encodes $\langle r, o\rangle$ into a single natural number $o' = \ulcorner\langle r, o\rangle\urcorner$. If $s$ is empty, then $e'$ returns reward $r' = 0$ and observation $o'$. Otherwise, $e'$ computes a modified version $t$ of $s$ by replacing every reward-observation pair $\ldots, r'_i, o'_i, \ldots$ in $s$ by the projections $\ldots r_i, o_i \ldots$ projected by $o'_i = \ulcorner\langle r_i, o_i\rangle\urcorner$. Finally, $e'$ lets $r' = 1$ if $T(t) = a$ (where $a$ is the last action in $s$), or $r' = -1$ otherwise, and outputs reward $r'$ and observation $o'$.*

In Example 14, one might imagine $e'$ as a room containing nothing but an arcade game $e$. There is nothing for the agent in the room to do except play this arcade game. When played, the arcade game visually displays rewards, but the agent merely observes them, and does not "feel" them. However, the agent is rewarded for acting as if actually feeling those displayed rewards, and punished for not so acting. In this way, the agent is incentivized to self-identify with the protagonist in the video-game, self-reflexively asking, "Which action would I take if I actually *felt* those rewards which I *see* on the screen?"

An abstract implementation of Example 14 is available in abstract/SelfInsert.py in [4], except that there, in order to avoid the encoding details, we give an implementation in which observations are allowed to literally be pairs.

## 5.2 Playing in the mirror

> "I may add that when a few days under nine months old he associated his own name with his image in the looking-glass, and when called by name would turn towards the glass even when at some distance from it."—Charles Darwin [15]

It has been suggested [23] that recognizing oneself in the mirror is linked to the development of certain parts of the human psychology. Using the techniques

developed so far, we can attempt to incentivize the RL agent to in some sense recognize itself in a mirror.

**Example 15.** *Suppose $e$ is a merciful environment whose observations encode snapshots of a room. Assume the room contains a mirror, and assume the room is laid out in such a way that everything important in the room is visible in the mirror (assume the environment constrains the agent to never look away from the mirror). We could derive an extended environment $e'$ which shows the same observations as $e$ and gives the same rewards, except it punishes the agent for acting differently than the agent would act if the agent* only *observes the mirror. To make this precise, for any $s = (r_0, o_0, a_0, \ldots, r_n, o_n) \in (\mathcal{ROA})^*\mathcal{RO}$ produced by $e'$, and any action $T(s) = a_n$, we would say that "$a_n$ is as if the agent only observes the mirror" if $T(s') = a_n$, where $s' = (r_0, o'_0, a_0, \ldots, r_n, o'_n)$, where each $o'_i$ is $o_i$ cropped to only the include the mirror.*

To make this even more elaborate, the $o'_i$ in the above example could be further modified by adding an image of the agent's "body" into the mirror. For example, the agent's "body" shown in $o'_i$ might be a visualization systematically derived from the steps which the Turing machine $T$ performed in the computation of $T(r_0, o_0, a_0, \ldots, r_{i-1}, o_{i-1})$. These computation steps would not be available to a traditional RL environment, but they are available to an extended environment because of the inclusion of $T$ itself as an argument passed to the extended environment.

## 5.3 Binocular vision

> "...as there are two eyes, so there may be in the soul something analogous, that of the eyes, doubtless, some one organ is formed, and hence their actualization in perception is one..."—Aristotle [6]

Humans seem to consciously perceive a three-dimensional model of their surrounding world, even though the raw data which we actually receive consists of two two-dimensional image-feeds (one for each eye). The following example is intended to incentivize an RL agent to learn to perceive the world through binocular vision like a human.

**Example 16.** *Suppose $V$ is a video game intended to be played on a virtual-reality headset, so at any moment during the game, $V$ produces two snapshots, one for the player's left eye, one for the player's right eye. Assume the player is constrained in $V$ so as never to be able to put their eyes into weird configurations (such as the weird configurations in [16]): thus, at any moment, the two snapshots $s_1, s_2$ which $V$ is displaying to the player are equivalent to a single 3-D matrix encoding the model $m(s_1, s_2)$ which the player is intended to perceive (with cells blanked out where the player's vision is obstructed by obstacles). Let $e$ be the extended environment whose observations encode pairs $\langle s_1, s_2 \rangle$ of snapshots displayed by $V$ in response to the player pressing keys encoded by the agent's actions. In response to history $(r_0, o_0, a_0, \ldots, r_n, o_n, a_n) \in (\mathcal{ROA})^*$*

*(where each $o_i$ encodes $\langle s_1^i, s_2^i \rangle$), let $r_{n+1} = 1$ if $a_n = T(r_0, o_0', a_0, \ldots, r_n, o_n')$, where each $o_i'$ encodes $m(s_1^i, s_2^i)$, otherwise let $r_{n+1} = -1$.*

In Example 16, upon being presented a sequence of pairs of 2-D snapshots, the agent is rewarded for acting as if it had instead been presented with an equivalent sequence of 3-D models. The agent is incentivized to self-reflectively ask, "Which action would I take if instead of observing those 2-D snapshot-pairs, I observed equivalent 3-D models?"

An abstract implementation of Example 16 (where the necessary camera functions are delegated to the user) is available in abstract/BinocularVision.py in [4].

## 5.4 Nature and Nurture

> "If only one soul was created, and all human souls are descended from it, who can say that he did not sin when Adam sinned?"— Augustine [7]

The following example is motivated by contemplating the possibility that maybe we all run the same software on some deep level.

**Example 17.** *(The Crying Baby) We define an extended environment $e$ as follows.*

- *Considered as actions taken by an adult (and also as observations seen by a baby), let $0$ denote "feed the baby" and let all naturals $> 0$ denote "don't feed the baby".*
- *Considered as actions taken by a baby (and also as observations seen by an adult), let $0$ denote "laugh" and let all naturals $> 0$ denote "cry".*
- *We define a* nutrition *function $N : (\mathcal{ROA})^* \to \mathbb{N}$ by $N(s) = 100 + 25f(s) - len(s)$ where $f(s)$ is the number of times that the action "feed the baby" is taken in $s$, and $len(s)$ is the length of $s$.*
- *Let $e(T, \langle\rangle) = \langle 1, \text{"laugh"} \rangle$.*
- *For each $\langle r_0, o_0, a_0, \ldots, r_n, o_n, a_n \rangle \in (\mathcal{ROA})^*$, let*

$$e(T, \langle r_0, o_0, a_0, \ldots, r_n, o_n, a_n \rangle) = \langle r, o \rangle$$

*where $r$ and $o$ are defined as follows. For each $i = 0, \ldots, n$, define*

$$r_i' = \begin{cases} 1 & \text{if } 50 \leq N(\langle r_0, o_0, a_0, \ldots, r_i, o_i, a_i \rangle) \leq 200, \\ -1 & \text{otherwise,} \end{cases}$$
$$o_i' = a_i$$
$$a_i' = T(\langle r_0', o_0', a_0', \ldots, r_i', o_i' \rangle).$$

*Define* $o = a'_n$ *and*

$$r = \begin{cases} 1 & \text{if } a'_n = \text{“laugh”}, \\ -1 & \text{if } a'_n = \text{“cry”}. \end{cases}$$

In Example 17, rather than mentally self-reflecting, the agent is physically self-reflected into the form of a newborn baby who has inherited the agent's own source-code. Despite having the same internal source-code $T$ (“nature”), the agent and the baby behave differently because they perceive the environment differently (“nurture”). The baby's objective is to maintain satiation within a satisfying range, whereas the agent's objective is to pacify the baby.

Using the same basic idea, one could extend Example 17 into an example modelling an entire society of interacting people of different types[9], all sharing the same internal “nature” ($T$).

Example 17 is implemented in CryingBaby.py at [4].

# 6 Examples involving self-recognition

**Example 18.** *(Incentivizing Self-recognition) Let* $s_0, s_1, \ldots$ *be a canonical computable enumeration of the nonempty sequences of* $(\mathcal{ROA})^*$*. We define an environment* $e$ *as follows:*

$$e(T, \langle\rangle) = \langle 0, s_0 \rangle$$
$$e(T, s \frown a) = \langle r, s_n \rangle$$

*where* $n = \frac{len(s \frown a)}{3}$ *is the number of actions in* $s \frown a$ *and where*

$$r = \begin{cases} 1 & \text{if } a > 0 \text{ and } a' = T(s'), \\ 1 & \text{if } a = 0 \text{ and } a' \neq T(s'), \\ -1 & \text{otherwise} \end{cases}$$

*where* $s_{n-1} = s' \frown a'$*.*

In Example 18, the agent is systematically shown all non-empty histories in $(\mathcal{ROA})^*$, and for each one, the agent either types “Looks like me” (any action $> 0$) or “Doesn't look like me” (0). When shown $s' \frown a'$, the agent is rewarded if and only if the agent correctly determines whether or not $a'$ is the action the agent itself would take in response to $s'$. Thus, the agent is incentivized to self-reflect in order to ask, “If I were prompted by that history, would I act that way?”

The following example is partly motivated by [33].

[9]Similar to how Plotinus compares society to an elaborate play where the actors “define their own rewards and punishments because they themselves assist in the rewards and punishments” [28].

**Example 19.** *(Recognizing other aspects of oneself) All the below environments are similar to Example 18, and we describe them informally to avoid technical details.*

- *(Supervised learning) Assume there is a canonical, computable function $f$ which transforms each RL agent $A$ into a supervised learning agent $f(A)$. By a* supervised learning trial *we mean a quadruple $\langle L, T, I, p\rangle$ where $L$ is a finite set of labels, $T$ is a sequence of images with labels from $L$ (a* training set*), $I$ is an unlabeled image, and $p : L \to \mathbb{Q} \cap [0, 1]$ is a function assigning to each label $\ell \in L$ a probability that $\ell$ is the correct label for $I$. We define an extended environment as follows. The agent $A$ is sytematically shown all supervised learning trials and must take action $> 0$ ("Looks like me") or $0$ ("Doesn't look like me"), and is rewarded or punished accordingly, depending whether or not $f(A)$ would output $p$ in response to $I$ after being trained with $T$.*

- *(Unsupervised learning) Assume there is a canonical, computable function $g$ which transforms each RL agent $A$ into an unsupervised learning agent $g(A)$. By an* unsupervised learning trial *we mean a triple $\langle n, D, C\rangle$ where $n$ is a positive integer, $D \subseteq \mathbb{Q}^n$ is a finite set of $n$-dimensional points with rational coordinates, and $C$ is a clustering of $D$. We define an extended environment as follows. The agent $A$ is systematically shown all unsupervised learning trials and must take action $> 0$ ("Looks like me") or $0$ ("Doesn't look like me"), and is rewarded or punished accordingly, depending whether or not $g(A)$ would cluster $D$ into clustering $C$.*

- *(The Turing Test) Assume there is a canonical, computable function $h$ which transforms each RL agent $A$ into an English-speaking chatbot $h(A)$. By a* chatbot trial *we mean a sequence of strings of English characters. We define an extended environment as follows. The agent $A$ is systematically shown all chatbot trials and must take action $> 0$ ("Looks like me") or $0$ ("Doesn't look like me"), and is rewarded or punished accordingly, depending whether or not even-numbered strings in the chatbot trial are what $h(A)$ would say in response to the user saying the odd-numbered strings.*

- *(Adversarial sequence prediction) Similar to the above environments, assuming a canonical computable function which transforms each RL agent into a predictor in the game of adversarial sequence prediction [20] [21].*

- *(Mechanical knowing agent) Assume there is a canonical, computable function $i$ which transforms each RL agent $A$ into a code $i(A)$ of a computably enumerable set of sentences in the language of Epistemic Arithmetic [30]; $i(A)$ is thought of as a mechanical knowing agent [11]. We define an extended environment as follows. The agent $A$ is systematically shown all sentences in the language of Epistemic Arithmetic, with each sentence being repeated infinitely often. Upon being shown sentence $\phi$ for the $n$th time, the agent must take action $> 0$ ("I know $\phi$ is true") or $0$ ("I'm not*

*sure if $\phi$ is true") and is rewarded if and only if $\phi$ is enumerated by $i(A)$ in $\leq n$ steps of computation.*

The extended environments in Example 19 incentivize the agent to self-reflect, asking itself questions like:

- "Is that how I would classify that image, given that training set?"
- "Is that how I would cluster that set of points?"
- "Is that what I would say in response to that conversation?"
- "Are those the adversarial sequence predictions I would make?"
- "Does that mechanical knowing agent know the same things I know?"

That perceptive non-RL agents such as supervised learning agents, unsupervised learning agents, etc., nevertheless might have some connection to RL-style reward mechanisms is suggested by Aristotle: "Whatever has a sense has the capacity for pleasure and pain and therefore has pleasant and painful objects present to it..." [5].

# 7 Measuring Self-awareness

As an application, self-awareness-incentivizing extended environments could be used to quantify RL agents' self-awareness. In order to measure the self-awareness of a specific agent (whose source-code is available to us), we could run the agent against a battery of extended environments that incentivize self-awareness, and see how the agent performs. Ideally, the battery of extended environments should be fairly large and contain many different types of environments, since an agent could perform poorly (resp. well) in a smaller set of environments by chance, despite being quite self-aware (resp. self-unaware) in general[10]. The larger and more representative the battery of extended environments, the better the measurement of self-awareness would be. One way to obtain a large battery of environments would be to start with a battery of non-extended environments, and a separate body of handicap extended environments (as in Section 4), and exhaustively apply all the different handicaps to all the different non-extended environments. In [4], in the file AwarenessBenchmark.py, we offer a function for doing exactly this.

This technique could be useful for obtaining empirical insight into questions about the self-awareness, or lack thereof, of various entities. For example, in order to get some empirical insight into the self-awareness, or lack thereof, of an NLP system like GPT-3 [12], one could twist that system into an RL agent (by means of template programming) and see how well said RL agent performs in extended environments that incentivize self-awareness.

[10] In the same way, computer programs can perform well on IQ tests despite being dumb [29].

It might even be possible to use extended environments to test the self-awareness of living creatures such as lab mice. At least it would be, if a given species of lab mice were entirely "nurture", as opposed to "nature". What we mean by a species being entirely "nurture" is that any two members of that species, if raised through identical life circumstances, would act similarly in identical environments. Thus, if a species were entirely nurture, and if two specimens were born in identical rooms, and experienced identical or near-identical lives[11], and were then placed in identical mazes, then they would perform identically. Given such a species, it would be possible to see how members perform in extended environments, albeit perhaps at great expense. One would have to carefully raise many similar specimens through identical or near-identical lives, diverging only at key points in order to compute the underlying $T$ action-function on different histories. By seeing how the underlying $T$ agent performs on well-chosen extended environments, we could get an idea of how self-aware those specimens were.

# References


[1] Igor Aleksander. The category of machines that become conscious. *Journal of Artificial Intelligence and Consciousness*, 7(1):3–13, 2020.

[2] Samuel Allen Alexander. Intelligence via ultrafilters: structural properties of some intelligence comparators of deterministic Legg-Hutter agents. *Journal of Artificial General Intelligence*, 10(1):24–45, 2019.

[3] Samuel Allen Alexander. The Archimedean trap: Why traditional reinforcement learning will probably not yield AGI. *Journal of Artificial General Intelligence*, 11(1):70–85, 2020.

[4] Samuel Allen Alexander, Michael Castaneda, Kevin Compher, and Oscar Martinez. ExtendedEnvironments. GitHub repository, https://github.com/semitrivial/ExtendedEnvironments, 2021.

[5] Aristotle. On the soul. In Jonathan Barnes et al., editors, *The Complete Works of Aristotle*. Princeton University Press, 1984.

[6] Aristotle. Sense and sensibilia. In Jonathan Barnes et al., editors, *The Complete Works of Aristotle*. Princeton University Press, 1984.

[7] Augustine. *On Free Choice of the Will*. Hackett, 1993.

[8] Marc G Bellemare, Yavar Naddaf, Joel Veness, and Michael Bowling. The arcade learning environment: An evaluation platform for general agents. *Journal of Artificial Intelligence Research*, 47:253–279, 2013.


[11] Possibly even while still in the womb. To quote Plato: "But it's hardly surprising you haven't heard of these athletics of the embryo. It's a curious subject, but I'd like to tell you about it" [26].

[9] Benjamin Beyret, José Hernández-Orallo, Lucy Cheke, Marta Halina, Murray Shanahan, and Matthew Crosby. The animal-AI environment: Training and testing animal-like artificial cognition. *Preprint*, 2019.

[10] Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. OpenAI gym. *Preprint*, 2016.

[11] Timothy J Carlson. Knowledge, machines, and the consistency of Reinhardt's strong mechanistic thesis. *Annals of Pure and Applied Logic*, 105(1-3):51–82, 2000.

[12] David Chalmers. GPT-3 and general intelligence. In Annette Zimmermann, editor, *Philosophers On GPT-3*. Daily Nous, 2020.

[13] François Chollet. On the measure of intelligence. *Preprint*, 2019.

[14] Karl Cobbe, Chris Hesse, Jacob Hilton, and John Schulman. Leveraging procedural generation to benchmark reinforcement learning. In *International conference on machine learning*, pages 2048–2056. PMLR, 2020.

[15] Charles Darwin. A biographical sketch of an infant. *Mind*, 2(7):285–294, 1877.

[16] Regan M Gallagher and Naotsugu Tsuchiya. Third-eye rivalry. *i-Perception*, 11(4), 2020.

[17] Vaibhav Gavane. A measure of real-time intelligence. *Journal of Artificial General Intelligence*, 4(1):31–48, 2013.

[18] Ben Goertzel. Patterns, hypergraphs and embodied general intelligence. In *The 2006 IEEE international joint conference on neural network proceedings*, pages 451–458. IEEE, 2006.

[19] José Hernández-Orallo and David L Dowe. Measuring universal intelligence: Towards an anytime intelligence test. *Artificial Intelligence*, 174(18):1508–1539, 2010.

[20] Bill Hibbard. Adversarial sequence prediction. In *ICAGI*, pages 399–403, 2008.

[21] Bill Hibbard. Measuring agent intelligence via hierarchies of environments. In *ICAGI*, pages 303–308, 2011.

[22] Evan Hubinger, Chris van Merwijk, Vladimir Mikulik, Joar Skalse, and Scott Garrabrant. Risks from learned optimization in advanced machine learning systems. *arXiv preprint arXiv:1906.01820*, 2019.

[23] Jacques Lacan. The mirror stage as formative of the function of the I. In *Écrits: A selection*, pages 1–6. Routledge, 2001.

[24] Shane Legg and Marcus Hutter. Universal intelligence: A definition of machine intelligence. *Minds and machines*, 17(4):391–444, 2007.

[25] Robert Nozick. Newcomb's problem and two principles of choice. In Nicholas Rescher, editor, *Essays in honor of Carl G. Hempel*, pages 114–146. Springer, 1969.

[26] Plato. Laws. In John M Cooper, Douglas S Hutchinson, et al., editors, *Plato: complete works*. Hackett Publishing, 1997.

[27] Plato. Republic. In John M Cooper, Douglas S Hutchinson, et al., editors, *Plato: complete works*. Hackett Publishing, 1997.

[28] Plotinus. On providence. In Lloyd P. Gerson, editor, *Plotinus: The Enneads*. Cambridge University Press, 2017.

[29] Pritika Sanghi and David L Dowe. A computer program capable of passing IQ tests. In *4th Intl. Conf. on Cognitive Science (ICCS'03), Sydney*, pages 570–575, 2003.

[30] Stewart Shapiro. Epistemic and intuitionistic arithmetic. In *Studies in Logic and the Foundations of Mathematics*, volume 113, pages 11–46. Elsevier, 1985.

[31] Craig Sherstan, Adam White, Marlos C Machado, and Patrick M Pilarski. Introspective agents: Confidence measures for general value functions. In *International Conference on Artificial General Intelligence*, pages 258–261. Springer, 2016.

[32] Christian Wirth, Riad Akrour, Gerhard Neumann, and Johannes Fürnkranz. A survey of preference-based reinforcement learning methods. *The Journal of Machine Learning Research*, 18(1):4945–4990, 2017.

[33] Roman V Yampolskiy. AI-complete, AI-hard, or AI-easy–classification of problems in AI. In *The 23rd Midwest Artificial Intelligence and Cognitive Science Conference*, 2012.

[34] Roman V Yampolskiy. Leakproofing the singularity-artificial intelligence confinement problem. *JCS*, 19, 2012.

[35] Yufan Zhao, Michael R Kosorok, and Donglin Zeng. Reinforcement learning design for cancer clinical trials. *Statistics in medicine*, 28(26):3294–3315, 2009.