Optimal Artificial Curiosity, Creativity, Music, and the Fine Arts

Jürgen Schmidhuber

TU Munich, Boltzmannstr. 3, 85748 Garching, München, Germany &
IDSIA, Galleria 2, 6928 Manno (Lugano), Switzerland
`juergen@idsia.ch - http://www.idsia.ch/~juergen`

Abstract. Even in absence of external reward, babies and scientists and others explore their world. Using some sort of adaptive predictive world model, they improve their ability to answer questions such as: what happens if I do this or that? They lose interest in both the predictable things and those predicted to remain unpredictable despite some effort. We can design curious robots that do the same. The author's basic idea for doing so (1990, 1991): a reinforcement learning (RL) controller is rewarded for action sequences that improve the predictor. Here we revisit this idea in the context of recent results on optimal predictors and optimal RL machines, proposing several new variants of the basic principle. Finally we point out how the fine arts can be formally understood as a consequence of the principle: given some subjective observer, great works of art and music yield observation histories exhibiting more novel, previously unknown compressibility / regularity / predictability (with respect to the observer's particular learning algorithm) than lesser works, thus deepening the observer's understanding of the world and what's possible in it.

1 Introduction

Consider a learning robotic agent with a single life which consists of discrete cycles or time steps $t = 1, 2, \ldots, T$. Its total lifetime T may or may not be known in advance. In what follows,the value of any time-varying variable Q at time t ($1 \leq t \leq T$) will be denoted by $Q(t)$, the ordered sequence of values $Q(1), \ldots, Q(t)$ by $Q(\leq t)$, and the (possibly empty) sequence $Q(1), \ldots, Q(t-1)$ by $Q(< t)$.

At any given t the robot receives a real-valued input vector $x(t)$ from the environment and executes a real-valued action $y(t)$ which may affect future inputs; at times $t < T$ its goal is to maximize future success or *utility*

$$u(t) = E_\mu \left[\sum_{\tau=t+1}^{T} r(\tau) \;\middle|\; h(\leq t) \right], \tag{1}$$

where $r(t)$ is an additional real-valued reward input at time t, $h(t)$ the ordered triple $[x(t), y(t), r(t)]$ (hence $h(\leq t)$ is the known history up to t), and $E_\mu(\cdot \mid \cdot)$ denotes the conditional expectation operator with respect to some possibly unknown distribution μ from a set M of possible distributions. Here M reflects whatever is known about the possibly probabilistic reactions of the environment. For example, M may contain all computable distributions [47, 48, 15, 9]. Note that unlike in most previous work by others [11, 50], but like in much of the author's own previous work [44, 34], there is just one life, no need for predefined repeatable trials, no restriction to Markovian interfaces between sensors and environment [25], and the utility function implicitly takes into account the expected remaining lifespan $E_\mu(T \mid h(\leq t))$ and thus the possibility to extend it through appropriate actions [34, 38–40].

Intuitively, to achieve its goal the robot may profit from spending some time on building a predictive world model, by exploring its environment and learning about the consequences of its actions in particular contexts. Such activity is commonly referred to as *curiosity*. The obtained world model may later speed up or otherwise facilitate the computation of action sequences provoking external rewards.

Recent work has led to the first learning machines that are universal and optimal in various very general senses [9, 38, 39]. Such machines can in principle find out by themselves whether curiosity and world model construction are useful or useless in a given environment, and learn to behave accordingly.

The present paper, however, will assume *a priori* that world model building is good and should be done; here we shall not worry about the possibility that "curiosity may kill the cat." Towards this end, in the spirit of our previous work [23, 22, 49, 29, 31], we split the reward signal $r(t)$ into two scalar real-valued components: $r(t) = g(r_{ext}(t), r_{int}(t))$, where g maps pairs of real values to real values, e.g., $g(a, b) = a + b$. Here $r_{ext}(t)$ denotes traditional *external* reward provided by the environment, such as pain (negative reward) in response to bumping against a wall, or pleasure (positive reward) in response to reaching some teacher-given goal state. In the context of the present paper, however, we are especially interested in $r_{int}(t)$, the internal reward, or intrinsic reward, or *curiosity* reward, which is provided whenever an internal predictive world model of the robot improves in some sense. In fact, we will initially focus on the case $r_{ext}(t) = 0$ for all valid t. The basic principle remains the one we published before [22, 23, 49, 29, 31, 35]:

Principle 1 *Generate curiosity reward for the adaptive action selector (or controller) in response to predictor improvements.*

So we conceptually separate the goal (understanding the world) from the means of achieving the goal. Once the goal is formally specified in terms of an algorithm for computing curiosity rewards, let the controller's RL mechanism figure out how to translate such rewards into world model-improving action sequences.

What kind of learning algorithm should the predictor use? How can we measure the predictor's improvements? Which RL algorithm should the controller use? In what follows, we will first briefly review previous work, then formalize a rather general framework (Section 3) into which we may plug various predictors (Section 3.1), measures of predictor performance (Section 3.2), measures of predictor performance improvement (Section 3.3), and RL agorithms for the controller. Subsequently we define *optimal* curiosity behavior, relative to the computational restrictions of some given predictor, and explain how to achieve it through plugging in recent, theoretically optimal, universal RL machines. Finally we will use the framework to give the first formal, technical definition of *good art* and *good music* relative to some subjective world model-building observer (previous explanations in the literature on fine arts and music were informal at best), and discuss illustrative examples.

2 Previous work

Our first publications on artificial curiosity [21, 24] (1990, 1991) described a predictor based on a recurrent neural network [53, 54, 20, 26, 17, 37] (in principle a rather powerful computational device, even by today's machine learning standards), predicting inputs $x(t)$ and $r(t)$ from the entire history of previous inputs and actions. The curiosity rewards were proportional to the predictor errors, that is, it was implicitly and optimistically assumed that the predictor will indeed improve whenever its error is high. Follow-up work from the same year [22, 23] pointed out that this approach may be inappropriate, especially in probabilistic environments: **We should not not focus on the errors of the predictor, but on its improvements.** Otherwise the system will concentrate its search on those parts of the environment where it can always get high prediction errors due to noise or randomness, or due to computational limitations of the predictor. While the neural predictor of the implementation described in the follow-up work was indeed computationally less powerful than the previous one [24], there was a novelty, namely, an explicit (neural) adaptive model of the predictor's improvements. This model essentially learned to predict the predictor's changes. For example, although noise details were unpredictable and led to wildly varying target signals for the predictor, in the long run these signals did not change the adaptive predictor parameters much, and the predictor of predictor changes was able to learn this. A standard RL algorithm [52, 11, 50] was fed with curiosity reward signals proportional to the expected long-term predictor changes, and thus tried to maximize information gain [5, 10, 16, 19, 4] within the given limitations. Additional follow-up work (1995) also focused on non-deterministic worlds [49].

More recent work [29, 31] (1997-2002) greatly increased the computational power of controller and predictor, implementing both as symmetric, opposing modules consisting of self-modifying probabilistic programs [44, 45] written in a universal programming language [6, 51]. The internal storage for temporary computational results of the programs was viewed as part of the changing environment. Each module could suggest experiments in the form of probabilistic algorithms to be executed, and make confident predictions about their effects, by betting on their outcomes, where the *"betting money"* essentially played the role of the intrinsic reward. The opposing module could reject or accept the bet in a zero-sum game, by making a contrary prediction. In case of acceptance the winner was determined by executing the algorithmic experiment and checking its outcome; the money was eventually transferred from the surprised loser to the confirmed winner. Both modules tried to maximize their money using a rather general RL algorithm designed for complex stochastic policies [44, 45].

All the references above also exemplified experimentally that the presence of curiosity reward $r_{int}(t)$ can speed up the collection of *external* reward.

Recently several researchers also implemented variants or approximations of Principle 1. Singh and Barto focused on an implementation within the option framework of RL [2, 46], directly using prediction errors as curiosity rewards. Additional implementations were presented at the recent 2005 AAAI Spring Symposium on Developmental Robotics [3].

In what follows, we will formulate a general framework which allows for discussing optimal predictors, optimal predictor improvements, and optimal RL algorithms for the controller.

3 General Synchronous Framework for Curiosity Reward

At any time t ($1 \leq t < T$), given some predictor or world model p and history $h(\leq t)$, let $C(p, h(\leq t))$ denote p's performance on $h(\leq t)$. Several more or less general types of predictor are discussed in Section 3.1, various natural performance measures C in Section 3.2. Let $p(t)$ denote the robot's current predictor, and $s(t)$ its current controller, and do:

1. Let $s(t)$ use (parts of) $h(\leq t)$ to select and execute $y(t+1)$.
2. Observe $x(t+1)$.
3. Evaluate $p(t)$ on (known parts of) $h(\leq t+1)$, to obtain $C(p(t), h(\leq t+1))$.
4. Let the predictor's learning algorithm use (known parts of) $h(\leq t+1)$ to obtain a hopefully better predictor $p(t+1)$.
5. Evaluate $p(t+1)$ on $h(\leq t+1)$, to obtain $C(p(t+1), h(\leq t+1))$.
6. Generate curiosity reward
$$r_{int}(t+1) = f[C(p(t+1), h(\leq t+1)), C(p(t), h(\leq t+1))] \tag{2}$$
in response to the predictor's progress between times t and $t+1$, where f maps pairs of real values to real values. Various alternative progress measures are discussed in Section 3.3; most obvious is $f(a, b) = a - b$.
7. Let the controller's RL algorithm use $h(\leq t+1)$, in particular, $r_{int}(t+1)$, and possibly also the new predictive world model $p(t+1)$ itself, to obtain a new controller $s(t+1)$, in line with objective (1). RL algorithms that are optimal in a certain sense will be discussed in Section 6.

The framework is labelled *"synchronous"* as it synchronizes action selection and reward-generation and learning in a fixed step-by-step process. This may be incovenient and actually unrealistic for practical purposes. For such reasons we will later (Section 5) discuss an asynchronous variant that loosens the strict coupling between the generation of curiosity reward and other system activities.

3.1 Predictors / World Models / History Compressors

The complexity of evaluating some predictor p on $h(\leq t)$ depends on both p and its performance measure C. Let us first focus on the former. Given t, one of the simplest p will just use a linear mapping to predict $x(t+1)$ from $x(t)$ and $y(t+1)$; see Section 4. More complex p such as adaptive recurrent neural networks (RNN) [53, 54, 20, 26, 17, 7, 41, 36, 37] will use a nonlinear mapping and possibly the entire history $h(\leq t)$ as a basis for the predictions. In fact, our first work on artificial curiosity [24] focused on online learning RNN of this type. A theoretically optimal predictor would be Solomonoff's universal induction scheme [47, 48, 15], to be discussed in more detail in Section 4.

Prediction and compression are closely related. A p that correctly predicts many $x(\tau)$, given $h(< \tau)$, for $1 \leq \tau \leq t$, can be used to encode $h(\leq t)$ compactly: Given p, only the wrongly predicted $x(\tau)$ plus information about the corresponding time steps τ are necessary to reconstruct $h(\leq t)$, e.g., [27]. Similarly, a p that learns a probability distribution of the possible next events, given previous events, can be used to efficiently encode observations with high (low) predicted probability by few (many) bits [8, 42], thus achieving a compressed history representation.

Generally speaking, we may view p as the essential part of a program that re-computes $h(\leq t)$. If this program is short in comparison to $h(\leq t)$, then $h(\leq t)$ is regular or non-random [47, 12, 15, 32], presumably reflecting essential environmental laws. Then p may also be highly useful for predicting future, yet unseen $x(\tau)$ for $\tau > t$.

3.2 Predictor Performance Measures

Given predictor p and time t, a naive predictor performance measure $C = C_{naive}$ will ignore all but the most recent event:

$$C_{naive}(p, h(\leq t)) = || \, pred(p, x(t)) - x(t) \, ||^2, \tag{3}$$

where $pred(p, x(\tau))$ is p's prediction of $x(\tau)$. Similar naive measures ignore all but a few recent observations in a sliding time window of events. A more computationally expensive performance measure re-evaluates p on *all* external inputs observed so far. The costs of this are linear in t, assuming p consumes equal amounts of computation time for each single prediction:

$$C_x(p, h(\leq t)) = \sum_{\tau=1}^{t} || \, pred(p, x(\tau)) - x(\tau) \, ||^2 \, . \tag{4}$$

A similar measure that also takes into account predictions of reward would be

$$C_{xr}(p, h(\leq t)) = C_x(p, h(\leq t)) + \sum_{\tau=1}^{t} || \, pred(p, r(\tau)) - r(\tau) \, ||^2 \, . \tag{5}$$

If the predictor also tries to predict future controller actions we obtain

$$C_{xry}(p, h(\leq t)) = C_{xr}(p, h(\leq t)) + \sum_{\tau=1}^{t} || \, pred(p, y(\tau)) - y(\tau) \, ||^2 \, . \tag{6}$$

Similar performance measures can be obtained for predictors p predicting a probability distribution of the next possible observations, given previous observations (Section 3.1), by replacing the mean squared error by statistical similarity measures, such as the Kullback-Liebler distance between two probability distributions [13].

If we view p as a program that compresses $h(\leq t)$ (Section 3.1), then an appropriate performance measure would be

$$C_l(p, h(\leq t)) = l(p), \tag{7}$$

where $l(p)$ denotes the length of p, measured in number of bits: the shorter p, the more algorithimic regularity and compressibility and predictability and lawfulness in the observations so far. The ultimate limit for $C_l(p, h(\leq t))$ would be $K^*(h(\leq t))$, a variant of the Kolmogorov complexity of $h(\leq t)$, namely, the length of the shortest program (for the given hardware) that computes an output starting with $h(\leq t)$ [47, 12, 15, 32].

Clearly, many other similar performance measures are possible. Later we will focus on the apparently most sound ones: those that re-evaluate p on the entire observation history so far. To our knowledge such measures have not yet been used in experimental work.

3.3 Predictor Performance Improvement Measures

The previous Section 3.2 just discussed measures of predictor performance, but not of performance *improvement*, which is the essential issue in our curiosity-oriented context. To repeat the point made in Section 2: **The important thing are the improvements of the predictor, not its errors.** The most obvious f for step 6 of the general framework above is $f(a, b) = a - b$. Our first work [24, 21] as well as work by Singh & Barto [2, 46] and others [3], however, essentially used $f(a, b) = b$, with $C = C_{naive}$ of eq. 3, implicitly assuming that high prediction error automatically implies predictor improvements. In stochastic environments this is generally not true, and even in deterministic environments it is generally not true due to limitations of the predictor and its learning algorithm. This motivated the follow-up work [22, 23, 49, 29, 31]—compare Section 2. However, no previous implementation really used mathematically justifiable performance measures such as C_x or C_{xy} (Section 3.2). All introduced major simplifying assumptions, ignoring potentially misleading effects due to online parameter changes.

4 Optimal Predictors

At time t the agent cannot know more about the world than $h(\leq t)$, the entire history of sensory perceptions and actions so far. Let us assume for the moment that computation time is not an issue. At each time step t we may then use a rather expensive performance measure such as C_{xr}, eq. (5), to compute $r_{int}(t)$ in step 6 of the general framework, using $f(a, b) = a - b$.

Optimal Linear Predictors Let $p(t)$ be the *optimal* linear predictor trying to map $x(\tau - 1), y(\tau)$ to $x(\tau)$ for $1 < \tau \leq t$ such that $C_x(p(t), h(\leq t))$ is minimized, eq. (4). We may find such an optimum using the well-known pseudo-inverse algorithm [**?**]. That is, we obtain a well-defined and easily computable curiosity reward $r_{int}(t)$.

We know that $r_{int}(t) \geq 0$ for all t, since the new predictor $p(t + 1)$ will never perform worse on $h(\leq t + 1)$ than $p(t)$.

The costs of computing the optimal $p(t)$ tend to grow polynomially in t. In simulated environments this is no major problem as we can wait with the t-specific update of the environment until $p(t)$ has been computed. Realistic environments, however, do not wait. Therefore, to avoid the computation of an optimal predictor at every single time step, Section 5 will discuss a natural asynchronous variant of the basic framework.

Optimal Universal Predictors Solomonoff's theoretically optimal universal predictors and their Bayesian learning algorithms [47, 48, 15, 9] only assume that the reactions of the environment are sampled from an unknown probability distribution μ contained in a set M of all enumerable distributions—compare text after equation (1). That is, given an observation sequence $q(\leq t)$, we only assume there exists a computer program that can compute the probability of the next possible $q(t+1)$, given $q(\leq t)$. Since we typically do not know this program, we predict using a mixture distribution ξ, a weighted sum of *all* distributions $\in \mathcal{M}$, where the sum of the weights does not exceed 1. It turns out that this is indeed the best one can possibly do, in a very general sense [48, 9]. The drawback is that the scheme is incomputable, since M contains infinitely many distributions.

We may increase the theoretical power of the scheme by augmenting M by certain non-enumerable but limit-computable distributions [32], or restrict it such that it becomes computable, e.g., by assuming the world is computed by some unknown but deterministic computer program sampled from the Speed Prior [33] which assigns low probability to environments that are hard to compute by any method.

Other Predictors Many alternative, less universal predictors can be defined, more general than the linear ones, less general than the universal ones, yet still optimal in some well-defined sense reflecting the predictor's computational limitations. For example, given cost function C_x and RNN predictor $p(t)$, it is formally clear what is an optimal $p(t)$. It is less clear how to find it efficiently, though. Given $h(\leq t)$, standard RNN algorithms [53, 54, 20, 26, 17, 7, 41, 36, 37] are not guaranteed to find an optimal RNN within reasonable time; they are based on gradient descent methods subject to local optima.

Practical applications will have to take into account such limitations of existing prediction algorithms. To facilitate their discussion, we will now introduce an asynchronous variant of the framework in Section 3.

5 General Asynchronous Framework for Curiosity Reward

The *synchronous* framework of Section 3 may be unnatural for practical implementations. For example, the costs of computing an optimal linear $p(t)$ (Section 4) based on a performance measure such as C_x, eq. (4), grow linearly in t. For large t, however, it is unrealistic to assume that we can evaluate $p(t)$ within a single time step. To further decouple the predictor's evaluation and learning procedures from those of the controller, we describe an asynchronous variant of the framework.

Controller: At any time t ($1 \leq t < T$) do:

1. Let $s(t)$ use (parts of) $h(\leq t)$ to select and execute $y(t+1)$.
2. Observe $x(t+1)$.
3. Check if there is non-zero curiosity reward $r_{int}(t+1)$ provided by the separate, asynchronously running predictor learning algorithm (see below). If not, set $r_{int}(t+1) = 0$.

4. Let the controller's RL algorithm use $h(\leq t+1)$ including $r_{int}(t+1)$ (and possibly also the latest available predictive world model provided by the predictor below) to obtain a new controller $s(t+1)$, in line with objective (1).

Predictor: Set p_{new} equal to the initial predictor. Starting at time 1, repeat forever until interrupted by death T:

1. Set $p_{old} = p_{new}$; get current time step t and set $h_{old} = h(\leq t)$.
2. Evaluate p_{old} on h_{old}, to obtain $C(p_{old}, h_{old})$ (Section 3.2). This may take many time steps.
3. Let the predictor's learning algorithm use h_{old} to obtain a hopefully better predictor p_{new}. Although this may take many time steps, p_{new} may not be optimal, due to limitations of the learning algorithm, e.g., local maxima.
4. Evaluate p_{new} on h_{old}, to obtain $C(p_{new}, h_{old})$. This may take many time steps.
5. Get current time step τ and generate curiosity reward

$$r_{int}(\tau) = f[C(p_{old}, h_{old}), C(p_{new}, h_{old})], \tag{8}$$

e.g., $f(a, b) = a - b$; see Section 3.3.

Clearly, this asynchronous scheme may cause long temporal delays between controller actions and corresponding curiosity rewards. This may further increase the burden on the controller's RL algorithm whose task is to assign credit to past actions (to inform the controller about beginnings of predictor evaluation processes etc., we may augment its input by unique representations of such events). Nevertheless, there are RL algorithms for this purpose which are theoretically optimal in various senses, to be discussed next.

6 Optimal Curiosity & Creativity

Our chosen predictor typically will have certain computational limitations. In absence of any external rewards, we may define *optimal pure curiosity behavior* relative to these limitations: At time t this behavior selects the action that maximizes

$$u(t) = E_{\mu}\left[\sum_{\tau=t+1}^{T} r_{int}(\tau) \;\middle|\; h(\leq t)\right]. \tag{9}$$

The resulting task of the controller's RL algorithm may be a formidable one, even when we are using very simple predictors and the synchronous framework of Section 3. For example, the optimal linear predictors (Section 4) may make it quite hard for the RL algorithm to compute action sequences that maximize future expected curiosity reward according to objective (9). As the system is revisiting previously unpredictable parts of the environment, some of those will tend to become more predictable, that is, the corresponding curiosity rewards will decrease over time. An optimal RL algorithm must somehow detect and then *predict* this decrease, and act accordingly. Traditional RL algorithms [11, 50], however, do not provide any theoretical guarantee of optimality for such situations. (This is not to say though that sub-optimal RL methods may not lead to limited success in certain applications; experimental studies might lead to interesting insights.)

Let us first make the natural assumption that the predictor is not super-complex such as Solomonoff's, that is, its output and and $r_{int}(t)$ are computable for all t. Is there an optimal RL algorithm for achieving objective (9)? Indeed, there is, for both the synchronuous and asynchronuous framework. Its drawback, however, is that itself is not computable in finite time. Nevertheless, it serves as a reference point for defining what's achievable at best.

6.1 Optimal But Incomputable Action Selector

At any time t, Hutter's recent theoretically optimal yet uncomputable RL algorithm AIXI [9] uses Solomonoff's universal prediction scheme (Section 4) to select those action sequences that promise maximal future reward up to some horizon, typically $2t$, given the current data $h(\leq t)$. We may adapt this to the case of any finite horizon T. That is, in cycle $t+1$, AIXI selects as next action the first in an action sequence maximizing ξ-predicted reward up to the horizon. Recent work [9] demonstrated AIXI 's optimal use of observations as follows. The Bayes-optimal policy p^{ξ} based on the mixture ξ is self-optimizing in the sense that its average utility value converges asymptotically for all $\mu \in \mathcal{M}$ to the optimal value achieved by the (infeasible) Bayes-optimal policy p^{μ} which knows μ in advance. The necessary condition that $\mathcal{M}$ admits self-optimizing policies is also sufficient. Furthermore, p^{ξ} is Pareto-optimal in the sense that there is no other policy yielding higher or equal value in *all* environments $\nu \in \mathcal{M}$ and a strictly higher value in at least one [9].

6.2 Computable Selector of Provably Optimal Actions, Given Current System

AIXI needs unlimited computation time. To take the consumed computation time into account in a general, optimal way, we may use the recent Gödel machines [34, 38–40] instead. Gödel machines represent the first class of mathematically rigorous, general, fully self-referential, self-improving, optimally efficient problem solvers. In particular, they are applicable to the problem embodied by objective (9).

The initial software $\mathcal{S}$ of such a Gödel machine contains an initial problem solver, e.g., one of Hutter's approaches [9] or some less general, typical sub-optimal method [11, 50]. Simultaneously, it contains an asymptotically optimal initial proof searcher based on an online variant of Levin's *Universal Search* [14], which is used to run and test *proof techniques*. The latter are programs written in a universal programming language implemented on the Gödel machinewithin $\mathcal{S}$, able to compute proofs concerning the system's own future performance, based on an axiomatic system $\mathcal{A}$ encoded in $\mathcal{S}$. $\mathcal{A}$ describes the formal *utility* function, in our case eq. (9), the hardware properties, axioms of arithmetics and probability theory and string manipulation etc, and $\mathcal{S}$ itself, which is possible without introducing circularity [34].

Inspired by Kurt Gödel's celebrated self-referential formulas (1931), the Gödel machine rewrites any part of its own code in a computable way through a self-generated executable program as soon as it has found a proof that the rewrite is *useful* according to objective (9). According to the Global Optimality Theorem [34, 38–40], such a self-rewrite is globally optimal—no local maxima!—since the self-referential code first had to prove that it is not useful to continue the proof search for alternative self-rewrites.

If there is no provably useful, globally optimal way of rewriting $\mathcal{S}$ at all, then humans will not find one either. But if there is one, then $\mathcal{S}$ itself can find and exploit it. Unlike previous *non*-self-referential methods based on hardwired proof searchers [9], Gödel machines not only boast an optimal *order* of complexity but can optimally reduce any slowdowns hidden by the $O()$-notation, provided the utility of such speed-ups is provable at all.

6.3 Consequences of Optimal Action Selecton

Now let us apply any optimal RL algorithm to curiosity rewards. The expected consequences are obvious: at time t the controller will do the best to select an action $y(t)$ that starts an action sequence expected to create observations yielding maximal expected predictor progress up until expected death T. In particular, ignoring issues of computation time, it will focus in the best possible way on things that are currently still unpredictable but will soon become predictable through additional learning. It will get bored by things that are predictable. It will also get bored by things that are currently unpredictable but will apparently remain unpredictable, given the experience so far, or where the costs of making them predictable exceed those of making other things predictable, etc.

Previous work [23, 22, 49, 29, 31] already discussed such effects, but not in the context of theoretically optimal ways of achieving them.

7 Music and the Fine Arts

Works of art and music do not seem to have an obvious purpose. Some even classify them as superfluous [18]. Others try to justify them through their social aspects, e.g., [1]. Undoubtedly, however, many derive pleasure and rewards from perceiving works of art, such as certain paintings, or songs. What exactly is the source of these rewards? Do they reflect some non-obvious, hidden usefulness of art? Why do certain observers perceive some artworks as being superior to others?

While previous attempts at describing what is satisfactory art or music were informal, the frameworks of Sections 3 and 5 permit the first *technical, formal* approach to answering such questions.

Any artificial or human observer must perceive art sequentially, and typically also actively, e.g., through a sequence of attention-shifting eye saccades or camera movements scanning a sculpture, or internal shifts of attention that filter and emphasize sounds made py a pianist, while surpressing background noise.

Different subjective observers with different sensory apparati and learning algorithms will prefer different input sequences. Hence any objective theory of what's good art must take the subjective observer as a parameter, to answer questions such as: Which action sequences should he select to maximize his pleasure? According to our curiosity reward framework he should select one that maximizes the number of quickly learnable regularities that are new, relative to his current knowledge and his (usually limited) way of incorporating or learning new data.

7.1 Music

For example, which song should some given observer select right now? Not the one he just heard ten times in a row. It became too predictable in the process. But also not the new weird one with the completely unfamiliar rhythm and tonality. It seems too irregular and contain too much arbitrariness and subjective noise. He should try a song that is unfamiliar enough to contain somewhat unexpected harmonies or melodies or beats etc., but familiar enough to allow for quickly recognizing the presence of a new learnable regularity in the sound stream. Sure, this song will get boring over time, but not yet.

The observer dependence is illustrated by the fact that Schönberg's twelve tone music is less popular than certain Bach tunes, presumably because its algorithmic structure is less obvious to many human observers. Those with a prior education about the basic concepts and objectives and constraints of twelve tone music, however, tend to appreciate Schönberg more than those without such an education.

All of this perfectly fits our frameworks from Section 3 and 5. The current predictor of a given subjective observer tries to compress (compare Section 3.1) his history of acoustic and other inputs where possible. The action selector tries to find actions that improve the predictor's performance on the history so far. Interesting are musical and other subsequences with previously unknown yet learnable types of regularities, because they lead to predictor improvements. Boring are patterns that seem arbitrary or random, or whose structure seems too hard to understand.

7.2 Visual Arts

Similar statements not only hold for other dynamic art including dances and movies, but also for static paintings or sculptures, which cause dynamic pattern sequences due to attention-shifting actions [43] of the observer.

For example, consider Figure 1, due to the author [28], reprinted with friendly permission by the journal *Leonardo*. It depicts a butterfly approaching a vase with a flower. The image to the left can be specified by very few bits of information; it is constructible through a very simple method or algorithm based on fractal circle patterns [28]. People who understand this algorithm tend to appreciate the drawing more than others. They realize how simple it is. The important, reward-generating process is the one leading from not being aware of the pattern to seeing it. This is not an immediate, all-or-nothing, binary process though—the typical human visual system has a lot of experience with circles, and even without formal explanation tends to realize that there is something special about this butterfly, noting that it is made of curves that somehow fit each other, that exhibit some sort of regularity, although few people are able to immediately see how exactly the drawing was made without being told.

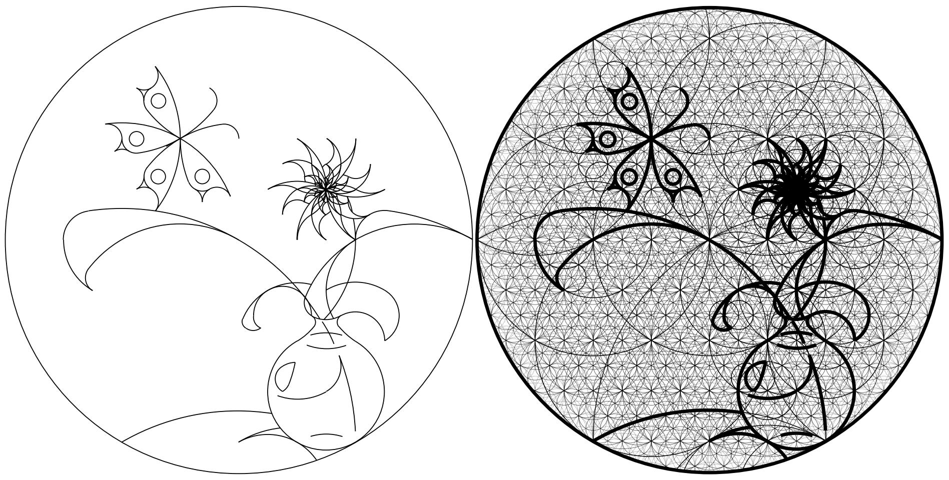

Fig. 1. Left: *Image of a butterfly approaching a vase with a flower, reprinted with friendly permission by the journal* Leonardo *[28].* **Right:** *Explanation of how the image was constructed through a very simple algorithm exploiting fractal circles [28]. The frame is a circle; its leftmost point is the center of another circle of the same size. Wherever two circles of equal size touch or intersect are centers of two more circles with equal and half size, respectively. Each line of the drawing is a segment of some circle, its endpoints are where circles touch or intersect. There are few big circles and many small ones. In general, the smaller a circle, the more bits are needed to specify it. The drawing to the left is simple (compressible) as it is based on few, rather large circles. Many human observers report that they derive a certain amount of pleasure from learning about this simplicity.*

7.3 Artists vs Observers: Any Difference?

So far we have focused on observers of artworks. What about the artists? Just like the observers get intrinsic rewards from observing artworks exhibiting new, previously unknown regularities, the artists get reward for making them. The distinction is not clear though. Artists can be observers and vice versa. Both artists and observers execute action sequences. The intrinsic motivations of both are fully compatible with our simple theoretical framework.

Some artists, however, crave for an additional source of reward besides the internal one provided by the discovery of a new work of art, namely, the external reward of *other* observers, who may praise them, or give them money, or both. Our framework, however, conceptually separates these two types of reward.

7.4 Beauty vs What's Interesting

In the 1990s we established a simple theory of subjective beauty [28, 30]: among several reprinted with patterns classified as "comparable" by some given subjective observer, the subjectively most beautiful is the one with the simplest (shortest) description, given the observer's particular method for encoding and memorizing it.

For example, mathematicians find beauty in a simple proof with a short description in the formal language they are using. Another example of beauty through simplicity is given by the construction of a geometrically simple, attractive face [30].

What's beautiful is not necessarily interesting though. A beautiful thing may be interesting (that is, trigger intrinsic curiosity rewards) only as long as it is new, that is, as long as the algorithmic regularity that makes it simple has not yet been learned and incorporated by the adaptive observer.

7.5 Summary: Art and Creativity as By-Products of Curiosity Rewards

In the light of the observations above, we postulate that active perception of all kinds of artworks and our interest therein is just a by-product of a curiosity reward-generating framework such as the ones of Section 3 or 5. These frameworks are sufficiently formal and precise to allow for computer implementations. The possible artificial observers obeying them will vary in terms of the computational power of their predictors and learning algorithms. This will influence what is good art to them, and what they find interesting.

In this sense we may indeed say that good observer-dependent art deepens the observer's insights about this world or possible worlds, connecting previously disconnected patterns in an initially surprising way that eventually becomes known and boring.

Is there a way of applying this scheme to the automatic creation of artworks that typical *human* observers will appreciate? To do it right would require a better understanding of the predictive mechanisms used by average humans. Popular artists may have acquired an intuitive understanding thereof, but more research is necessary to automate the process of creating widely popular art.

8 Conclusions

There are theoretically optimal ways of improving the predictive world model of a robotic agent. They are based on optimal reinforcement learners maximizing expected future reward. The rewards are the predictor's improvements on the observation history so far. They encourage the reinforcement learner to produce action sequences that cause the creation and the learning of new, previously unknown regularities in the sensory input stream. Art and creativity can be explained as by-products of such intrinsic curiosity rewards.

References

1. M. Balter. Seeking the key to music. *Science*, 306:1120–1122, 2004.
2. A. G. Barto, S. Singh, and N. Chentanez. Intrinsically motivated learning of hierarchical collections of skills. In *Proceedings of International Conference on Developmental Learning (ICDL)*. MIT Press, Cambridge, MA, 2004.
3. D. Blank and L. Meeden. Developmental Robotics AAAI Spring Symposium, Stanford, CA, 2005. http://cs.brynmawr.edu/DevRob05/schedule/.

4. D. A. Cohn. Neural network exploration using optimal experiment design. In J. Cowan, G. Tesauro, and J. Alspector, editors, *Advances in Neural Information Processing Systems 6*, pages 679–686. Morgan Kaufmann, 1994.
5. V. V. Fedorov. *Theory of optimal experiments*. Academic Press, 1972.
6. K. Gödel. Über formal unentscheidbare Sätze der Principia Mathematica und verwandter Systeme I. *Monatshefte für Mathematik und Physik*, 38:173–198, 1931.
7. S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1735–1780, 1997.
8. D. A. Huffman. A method for construction of minimum-redundancy codes. *Proceedings IRE*, 40:1098–1101, 1952.
9. M. Hutter. *Universal Artificial Intelligence: Sequential Decisions based on Algorithmic Probability*. Springer, Berlin, 2004. (On J. Schmidhuber's SNF grant 20-61847).
10. J. Hwang, J. Choi, S. Oh, and R. J. Marks II. Query-based learning applied to partially trained multilayer perceptrons. *IEEE Transactions on Neural Networks*, 2(1):131–136, 1991.
11. L. P. Kaelbling, M. L. Littman, and A. W. Moore. Reinforcement learning: a survey. *Journal of AI research*, 4:237–285, 1996.
12. A. N. Kolmogorov. Three approaches to the quantitative definition of information. *Problems of Information Transmission*, 1:1–11, 1965.
13. S. Kullback. *Statistics and Information Theory*. J. Wiley and Sons, New York, 1959.
14. L. A. Levin. Universal sequential search problems. *Problems of Information Transmission*, 9(3):265–266, 1973.
15. M. Li and P. M. B. Vitányi. *An Introduction to Kolmogorov Complexity and its Applications (2nd edition)*. Springer, 1997.
16. D. J. C. MacKay. Information-based objective functions for active data selection. *Neural Computation*, 4(2):550–604, 1992.
17. B. A. Pearlmutter. Gradient calculations for dynamic recurrent neural networks: A survey. *IEEE Transactions on Neural Networks*, 6(5):1212–1228, 1995.
18. S. Pinker. *How the mind works*. 1997.
19. M. Plutowski, G. Cottrell, and H. White. Learning Mackey-Glass from 25 examples, plus or minus 2. In J. Cowan, G. Tesauro, and J. Alspector, editors, *Advances in Neural Information Processing Systems 6*, pages 1135–1142. Morgan Kaufmann, 1994.
20. A. J. Robinson and F. Fallside. The utility driven dynamic error propagation network. Technical Report CUED/F-INFENG/TR.1, Cambridge University Engineering Department, 1987.
21. J. Schmidhuber. Making the world differentiable: On using fully recurrent self-supervised neural networks for dynamic reinforcement learning and planning in non-stationary environments. Technical Report FKI-126-90, Institut für Informatik, Technische Universität München, 1990.
22. J. Schmidhuber. Adaptive curiosity and adaptive confidence. Technical Report FKI-149-91, Institut für Informatik, Technische Universität München, April 1991. See also [23].
23. J. Schmidhuber. Curious model-building control systems. In *Proceedings of the International Joint Conference on Neural Networks, Singapore*, volume 2, pages 1458–1463. IEEE press, 1991.
24. J. Schmidhuber. A possibility for implementing curiosity and boredom in model-building neural controllers. In J. A. Meyer and S. W. Wilson, editors, *Proc. of the International Conference on Simulation of Adaptive Behavior: From Animals to Animats*, pages 222–227. MIT Press/Bradford Books, 1991.
25. J. Schmidhuber. Reinforcement learning in Markovian and non-Markovian environments. In D. S. Lippman, J. E. Moody, and D. S. Touretzky, editors, *Advances in Neural Information Processing Systems 3 (NIPS 3)*, pages 500–506. Morgan Kaufmann, 1991.
26. J. Schmidhuber. A fixed size storage $O(n^3)$ time complexity learning algorithm for fully recurrent continually running networks. *Neural Computation*, 4(2):243–248, 1992.

27. J. Schmidhuber. Learning complex, extended sequences using the principle of history compression. *Neural Computation*, 4(2):234–242, 1992.
28. J. Schmidhuber. Low-complexity art. *Leonardo, Journal of the International Society for the Arts, Sciences, and Technology*, 30(2):97–103, 1997.
29. J. Schmidhuber. What's interesting? Technical Report IDSIA-35-97, IDSIA, 1997. ftp://ftp.idsia.ch/pub/juergen/interest.ps.gz; extended abstract in Proc. Snowbird'98, Utah, 1998; see also [31].
30. J. Schmidhuber. Facial beauty and fractal geometry, 1998. Published in the Cogprint Archive: http://cogprints.soton.ac.uk.
31. J. Schmidhuber. Exploring the predictable. In A. Ghosh and S. Tsuitsui, editors, *Advances in Evolutionary Computing*, pages 579–612. Springer, 2002.
32. J. Schmidhuber. Hierarchies of generalized Kolmogorov complexities and nonenumerable universal measures computable in the limit. *International Journal of Foundations of Computer Science*, 13(4):587–612, 2002.
33. J. Schmidhuber. The Speed Prior: a new simplicity measure yielding near-optimal computable predictions. In J. Kivinen and R. H. Sloan, editors, *Proceedings of the 15th Annual Conference on Computational Learning Theory (COLT 2002)*, Lecture Notes in Artificial Intelligence, pages 216–228. Springer, Sydney, Australia, 2002.
34. J. Schmidhuber. Gödel machines: self-referential universal problem solvers making provably optimal self-improvements. Technical Report IDSIA-19-03, arXiv:cs.LO/0309048, IDSIA, Manno-Lugano, Switzerland, 2003.
35. J. Schmidhuber. Overview of artificial curiosity and active exploration, with links to publications since 1990, 2004. http://www.idsia.ch/˜juergen/interest.html.
36. J. Schmidhuber. Overview of work on robot learning, with publications, 2004. http://www.idsia.ch/˜juergen/learningrobots.html.
37. J. Schmidhuber. RNN overview, with links to a dozen journal publications, 2004. http://www.idsia.ch/˜juergen/rnn.html.
38. J. Schmidhuber. Completely self-referential optimal reinforcement learners. In W. Duch, J. Kacprzyk, E. Oja, and S. Zadrozny, editors, *Artificial Neural Networks: Biological Inspirations - ICANN 2005, LNCS 3697*, pages 223–233. Springer-Verlag Berlin Heidelberg, 2005. Plenary talk.
39. J. Schmidhuber. Gödel machines: fully self-referential optimal universal problem solvers. In B. Goertzel and C. Pennachin, editors, *Artificial General Intelligence*. Springer Verlag, in press, 2005.
40. J. Schmidhuber. Gödel machines: Towards a technical justification of consciousness. In D. Kudenko, D. Kazakov, and E. Alonso, editors, *Adaptive Agents and Multi-Agent Systems III (LNCS 3394)*, pages 1–23. Springer Verlag, 2005.
41. J. Schmidhuber and B. Bakker. NIPS 2003 RNNaissance workshop on recurrent neural networks, Whistler, CA, 2003. http://www.idsia.ch/˜juergen/rnnaissance.html.
42. J. Schmidhuber and S. Heil. Sequential neural text compression. *IEEE Transactions on Neural Networks*, 7(1):142–146, 1996.
43. J. Schmidhuber and R. Huber. Learning to generate artificial fovea trajectories for target detection. *International Journal of Neural Systems*, 2(1 & 2):135–141, 1991.
44. J. Schmidhuber, J. Zhao, and N. Schraudolph. Reinforcement learning with self-modifying policies. In S. Thrun and L. Pratt, editors, *Learning to learn*, pages 293–309. Kluwer, 1997.
45. J. Schmidhuber, J. Zhao, and M. Wiering. Shifting inductive bias with success-story algorithm, adaptive Levin search, and incremental self-improvement. *Machine Learning*, 28:105–130, 1997.
46. S. Singh, A. G. Barto, and N. Chentanez. Intrinsically motivated reinforcement learning. In *Advances in Neural Information Processing Systems 17 (NIPS)*. MIT Press, Cambridge, MA, 2005.

47. R. J. Solomonoff. A formal theory of inductive inference. Part I. *Information and Control*, 7:1–22, 1964.
48. R. J. Solomonoff. Complexity-based induction systems. *IEEE Transactions on Information Theory*, IT-24(5):422–432, 1978.
49. J. Storck, S. Hochreiter, and J. Schmidhuber. Reinforcement driven information acquisition in non-deterministic environments. In *Proceedings of the International Conference on Artificial Neural Networks, Paris*, volume 2, pages 159–164. EC2 & Cie, 1995.
50. R. Sutton and A. Barto. *Reinforcement learning: An introduction*. Cambridge, MA, MIT Press, 1998.
51. A. M. Turing. On computable numbers, with an application to the Entscheidungsproblem. *Proceedings of the London Mathematical Society, Series 2*, 41:230–267, 1936.
52. C. J. C. H. Watkins. *Learning from Delayed Rewards*. PhD thesis, King's College, Oxford, 1989.
53. P. J. Werbos. Generalization of backpropagation with application to a recurrent gas market model. *Neural Networks*, 1, 1988.
54. R. J. Williams and D. Zipser. Gradient-based learning algorithms for recurrent networks and their computational complexity. In *Back-propagation: Theory, Architectures and Applications*. Hillsdale, NJ: Erlbaum, 1994.