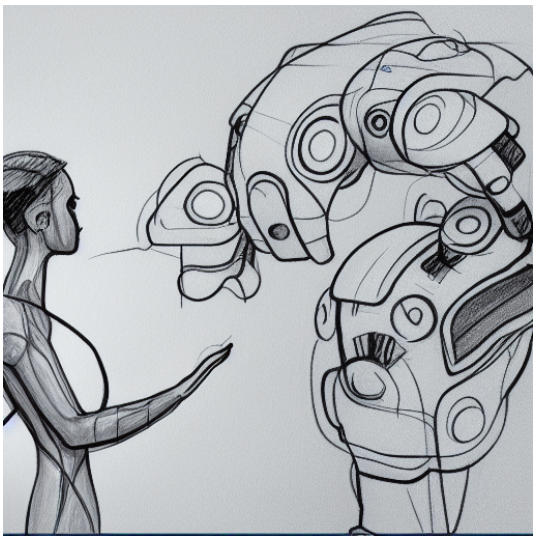
Towards Interactive Social Artificial Agents

Formation and Exploitation of Cultural Models in Autonomous Artificial Agents



By **Tristan Karch**

Under the supervision of Pierre-Yves Oudeyer & Clément Moulin-Frier

In partial fulfillment of the requirements

for the degree of Doctor of Philosophy

University of Bordeaux

Graduate school of Mathematics and Computer Science

Major in Computer Science

Submitted on X. Defended on Y.

Composition of the jury:

Contents

Acknowledgments

Abstract

Glossary

Autotelic from the Greek *auto* (self) and *telos* (end, goal), characterizes agents that generate their own goals and learning signals. In is equivalent to *intrinsically motivated and goal-conditioned*. 3

Cultural Model element of language or social interaction from which meaning can be derived. 3

Chapter 1

Introduction

One fundamental goal of Artificial Intelligence (AI) is to design embodied autonomous interactive agents that can evolve in various environments and complete a wide range of tasks. To that end, researchers in AI take several angles of attack and rely on different paradigms that consider different drivers for learning. In Reinforcement Learning (RL) (Sutton & Barto, 2018), agents learn from *exploration* of their environment. They rely solely on their experience of the world in order to solve a pre-defined task. In Imitation Learning (IL) (Pomerleau, 1991), agents learn from *demonstrations*, i.e. trajectories provided by an expert that correspond to the transitions required to take to solve a pre-defined task. In Multi-Agent Learning (Littman, 1994), agents learn in *cooperation* and need to interact with each other in order to solve collaborative tasks.

Recent extensions of RL algorithm have shown success in solving a wealth of problems such as playing the Atari videogames at super-human levels (Mnih et al., 2015), beating chess and go world champions (Silver et al., 2016), controlling stratospheric baloons (Bellemare et al., 2020) or even maintaining plasma in fusion reactors (Degrave et al., 2022). Similarly, IL methods coupled to Transformers (Vaswani et al., 2017) have enabled the training of a generalist agent on a massive dataset of diverse interactions (Reed et al., 2022). It has also been used to perform in-context reinforcement learning via algorithm distillation (Laskin et al., 2022). Finally, multi-agent methods have permitted populations of agents to play hide and seek (Baker et al., 2020) or even to collaboratively solve common-pool resource problems (Pérolat et al., 2017).

But unlike humans, these algorithms are still heavily sample-inefficient, requiring billions of transitions to become proficient on isolated tasks. Most importantly, they lack the ability to generalize and transfer across a wide variety of problems, to be creative, and tackle tasks never seen during training. They are far from displaying human-like capabilities in terms of open-ended learning. This is, perhaps, because they rely on isolated signals for learning. The way forward might be to build on child development theory and to consider learning from *sociocultural interactions*. Indeed humans are social beings, they interact and cooperate with their peers (Tomasello, 1999b; Tomasello et al., 2005; Brewer et al., 2014). As soon as they discover and learn a language, they assimilate thousands of years of experience embedded in their culture (Bruner, 1991). Most of their skills could not be learned in isolation. Formal education teaches them to reason systematically, books teach them history, and YouTube might teach them how to cook.

Most importantly, humans' values, traditions, norms, and most of their goals are cultural in essence.

The present research proposes to immerse artificial agents in social contexts in order to observe the impact of sociocultural interactions on learning. As displayed in figure 1.1, it has a dual objective. In the first part of this manuscript, we propose to use artificial agents as an anthropological tool to study the formation of cultural conventions in populations of individuals. More specifically, we investigate the key mechanisms required for the self-organization of cultural models between artificial agents in absence of pre-existing conventions. In the second part, we focus on autonomous artificial agents exploiting already existing cultural conventions to augment their capabilities in the open-ended skill acquisition problem. To accomplish this, we build on previous theories at the intersection of developmental psychology and machine learning to introduce a new framework coined *Vygotskian Artificial Intelligence* which enables sociocultural interactions to transform agents' learning signal, yielding better learners.

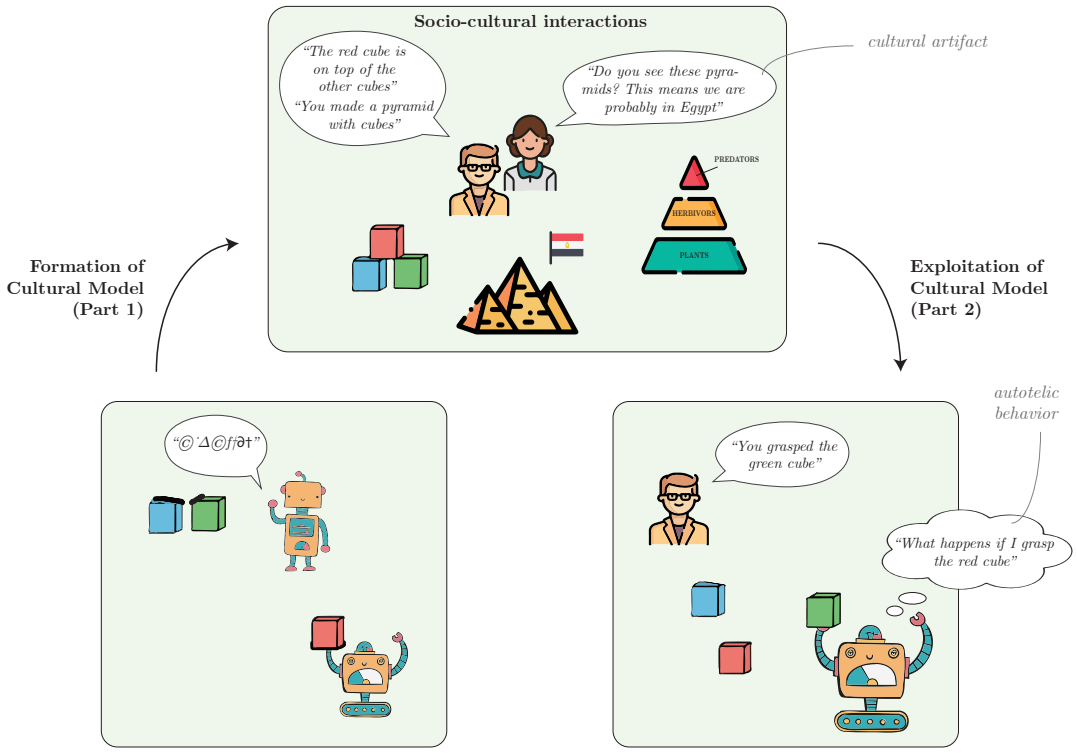


Figure 1.1: **Dual organization of the present research.** [...]

The remaining of this introduction presents key features of human learning that enable us to define the important notions of "autotelic learning" and "cultural convention" at the center of this research.

1.1 Humans are goal-directed social learners

Humans are an incredible source of inspiration for AI. They are the fastest learning system we can ever witness. Within only a few years, children learn to crawl and navigate their

home, to identify and manipulate objects, they even learn to speak and interact with their peers. How do they reach such a level of proficiency in such a short period of time?

Humans are autotelic learners

A central aspect of human development is the notion of goal. Elliot & Fryer (2008) propose the following goal construct definition:

> *A goal is a cognitive representation of a future object that the organism is committed to approach or avoid* (Elliot & Fryer, 2008).

intrinsic motivation (Piaget?)

In particular, children's exploration seems to be driven by intrinsically motivated brain processes that trigger spontaneous exploration for the mere purpose of experiencing novelty, surprise or learning progress (Gopnik et al., 1999; Kaplan & Oudeyer, 2007; Kidd & Hayden, 2015b). During exploratory play, children can also invent and pursue their own problems [19].

Csikzentmihalyi (1997) uses the term autotelic in his theory of flow to describe those agents or activities that are intrinsically motivated.

Steels will later use the term to designate [insert quote from steel]

Definition

Autotelic: from the Greek *auto* (self) and *telos* (end, goal), characterizes agents that generate their own goals and learning signals. In is equivalent to *intrinsically motivated and goal-conditioned.*

Humans are social learners

Humans are social beings; intrinsically motivated to interact and cooperate with their peers (Tomasello, 1999b; Tomasello et al., 2005; Brewer et al., 2014).

This knowledge, and some argue, some of our highest cognitive functions such as abstraction, compositional imagination, or relational thinking, are formed through linguistic and cultural interactions with others

For Vygotsky, linguistic social interactions such as descriptions, explanations, corrections, or play start as interpersonal processes before they are turned into intrapersonal cognitive processes through the process of internalization. 33–35 Following his vision, many psychologists,36–38 linguists,39–41, and philosophers42–45 argued for the importance of socio-cultural interactions in the development of human intelligence.

Definition

Cultural Model: element of language or social interaction from which meaning can be derived.

Towards Interactive Social Autonomous Agents

Drawing inspiration from children

Not necessarily a shift of paradigm but rather using social interactions (language and culture rachets) inside previous paradigms:

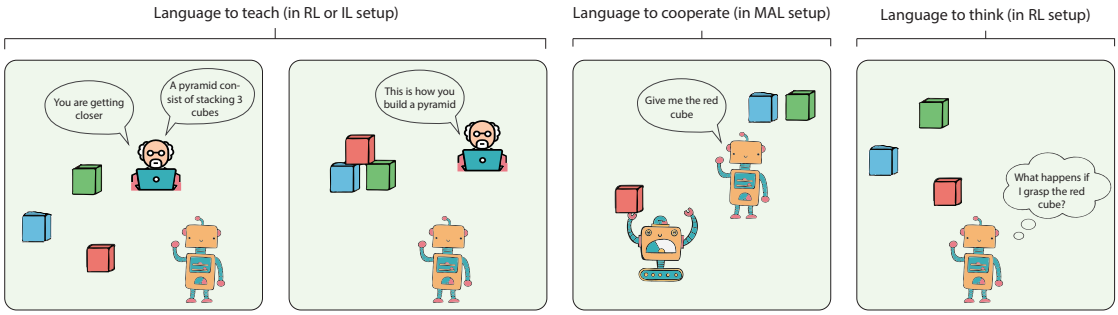


Figure 1.2

Inside social interaction is language and culture rachets they can be used to:

- Language to cooperate
- Language to teach Sigaud et al. (2021a)
- Language to organize thoughts

Objectives and Contributions

The end goal of the present research is to make progress towards the autonomous acquisition of repertoires of skills.

We advocate for socio-cultural interactions as the driver of learning as such the

This manuscript is organized around two main questions:

1. How can such a cultural model emerge in populations of agents?
2. How can agents leverage existing cultural models to become better learners?

There is no open-endedness without culture. From an engineering perspective, building open-ended environments for artificial agents requires a community effort.

Culture as youtube videos (from perspective)

Citation (bruner, vygotsky)

Chronological order of publications

Collaborations

Chapter 2

Background

Contents

This chapter introduces foundational concepts required for both parts of the manuscript

2.1 Concepts and Scope

2.1.1 Intrinsic Motivations

2.1.2 Cultural Models

2.1.3 Emergence of cultural models

2.1.4 The role of cultural models in human development

2.2 Tools and Frameworks

2.2.1 Reinforcement Learning

This sections presents background information on the RL problem, the multi-goal RL problem and the families of algorithms used to solve them. This will serve as a foundation to define the *intrinsically motivated skill acquisition problem* and introduce the

RL-based intrinsically motivated goal exploration process framework to solve it (RL-IMGEP, Section 6.2.1).

In a reinforcement learning (RL) problem, the agent learns to perform sequences of actions in an environment so as to maximize some notion of cumulative reward Sutton & Barto (2018). RL problems are commonly framed as Markov Decision Processes (MDPs): $\mathcal{M} = \{\mathcal{S}, \mathcal{A}, \mathcal{T}, \rho_0, R\}$ Sutton & Barto (2018). The agent and its environment, as well as their interaction dynamics are defined by the first components $\{\mathcal{S}, \mathcal{A}, \mathcal{T}, \rho_0\}$, where $s \in \mathcal{S}$ describes the current state of the agent-environment interaction and ρ_0 is the distribution over initial states. The agent can interact with the environment through actions $a \in \mathcal{A}$. Finally, the dynamics are characterized by the transition function $\mathcal{T}$ that dictates the distribution of the next state s' from the current state and action $\mathcal{T}(s' \mid s, a)$.

The objective of the agent in this environment is defined by the remaining component of the MDP: R. R is the reward function, it computes a reward for any transition: $R(s, a, s')$. Note that, in a traditional RL problem, the agent only receives the rewards corresponding to the transitions it experiences but does not have access to the function itself. The objective of the agent is to maximize the cumulative reward computed over complete episodes. When computing the aggregation of rewards, we often introduce discounting and give smaller weights to delayed rewards. R_t^{tot} is then computed as $R_t^{\text{tot}} = \sum_{i=t}^{\infty} \gamma^{i-t} R(s_{i-1}, a_i, s_i)$ with γ being a constant discount factor in $]0, 1]$. Each instance of an MDP implements an RL problem, also called a *task*.

Defining *Goals* for Reinforcement Learning

This section takes inspiration from the notion of *goal* in psychological research to inform the formalization of *goals* for reinforcement learning.

***Goals* in psychological research.** Working on the origin of the notion *goal* and its use in past psychological research, Elliot & Fryer (2008) propose a general definition:

> *A goal is a cognitive representation of a future object that the organism is committed to approach or avoid* Elliot & Fryer (2008).

Because goals are *cognitive representations*, only animate organisms that represent goals qualify as goal-conditioned. Because this representation relates to a *future object*, goals are cognitive imagination of future possibilities: goal-conditioned behavior is proactive, not reactive. Finally, organisms *commit* to their goal, their behavior is thus influenced directly by this cognitive representation.

Generalized *goals* for reinforcement learning. RL algorithms seem to be a good fit to train such goal-conditioned agents. Indeed, RL algorithms train learning agents (*organisms*) to maximize (*approach*) a cumulative (*future*) reward (*object*). In RL, goals can be seen as a set of *constraints* on one or several consecutive states that the agent seeks to respect. These constraints can be very strict and characterize a single target point in the state space (e.g. image-based goals) or a specific sub-space of the state space (e.g. target x-y coordinate in a maze, target block positions in manipulation tasks). They can also be more general, when expressed by language for example (e.g. '*find a red object or a wooden one*').

To represent these goals, RL agents must be able to 1) have a compact representation of them and 2) assess their progress towards it. This is why we propose the following formalization for RL goals: each goal is a $g = (z_g, R_g)$ pair where z_g is a compact *goal parameterization* or *goal embedding* and R_g is a *goal-achievement* function measuring progress towards the goal. The set of goal-achievement function can be represented as a single *goal-parameterized* or *goal-conditioned* reward function such that $R_{\mathcal{G}}(\cdot \mid z_g) = R_g(\cdot)$. With this definition we can express a diversity of goals, see Section 6.2.3 and Table 6.1.

The goal-achievement function and the goal-conditioned policy both assign *meaning* to a goal. The former defines what it means to achieve the goal, it describes how the world looks like when it is achieved. The latter characterizes the process by which this goal can be achieved; what the agent needs to do to achieve it. In this search for the meaning of a goal, the goal embedding can be seen as the map: the agent follows this map and via the two functions above, experiences the meaning of the goal.

Generalized definition of the goal construct for RL:

- ***Goal***: a $g = (z_g, R_g)$ pair where z_g is a compact *goal parameterization* or *goal embedding* and R_g is a *goal-achievement* function.
- **Goal-achievement function:** $R_g(\cdot) = R_{\mathcal{G}}(\cdot \mid z_g)$ where $R_{\mathcal{G}}$ is a goal-conditioned reward function.

The Multi-Goal Reinforcement Learning Problem

By replacing the unique reward function R by the space of reward functions $\mathcal{R}_{\mathcal{G}}$, RL problems can be extended to handle multiple goals: $\mathcal{M} = \{\mathcal{S}, \mathcal{A}, \mathcal{T}, \rho_0, \mathcal{R}_{\mathcal{G}}\}$. The term *goal* should not be mistaken for the term *task*, which refers to a particular MDP instance. As a result, *multi-task* RL refers to RL algorithms that tackle a set of MDPs that can differ by any of their components (e.g. $\mathcal{T}, R, \mathcal{S}_0$, etc.). The *multi-goal* RL problem can thus be seen as the particular case of the multi-task RL problem where MDPs differ by their reward functions. In the standard multi-goal RL problem, the set of goals — and thus the set of reward functions — is pre-defined by engineers. The experimenter sets goals to the agent, and provides the associated reward functions.

Solving the RL Problem with RL Algorithms and Related Approaches

The RL problem can be tackled by several types of optimization methods. In this survey, we focus on RL algorithms, as they currently demonstrate stronger capacities in multi-goal problems Florensa et al. (2018); Eysenbach et al. (2019); Warde-Farley et al. (2019); Pong et al. (2020); Lynch & Sermanet (2020); Hill et al. (2020d, 2021); Abramson et al. (2020); Colas et al. (2020b); Stooke et al. (2021).

RL algorithms use transitions collected via interactions between the agent and its environment $(s, a, s', R(s, a, s'))$ to train a *policy* π: a function generating the next action a based on the current state s so as to maximize a cumulative function of rewards.

Deep RL (DRL) is the extension of RL algorithms that leverage deep neural networks as function approximators to represent policies, reward and value functions. It has been powering most recent breakthrough in RL Eysenbach et al. (2019); Warde-Farley et al. (2019); Florensa et al. (2018); Pong et al. (2020); Lynch & Sermanet (2020); Hill et al. (2020d, 2021); Abramson et al. (2020); Colas et al. (2020b); Stooke et al. (2021).

Other sets of methods can also be used to train policies. Imitation Learning (IL) leverages demonstrations, i.e. transitions collected by another entity Ho & Ermon (2016); Hester et al. (2018). Evolutionary Computing (EC) is a group of population-based approaches where populations of policies are trained to maximize cumulative rewards using episodic samples Sehnke et al. (2010); Lehman & Stanley (2011); Wierstra et al. (2014); Mouret & Clune (2015); Salimans et al. (2017); Forestier et al. (2022); Colas et al. (2020d). Finally, in model-based RL approaches, agents learn a model of the transition function $\mathcal{T}$. Once learned, this model can be used to perform planning towards reward maximization or train a policy via RL using imagined samples (e.g. Schmidhuber (1990); Dayan et al. (1995); Nguyen-Tuong & Peters (2011); Chua et al. (2018); Charlesworth & Montana (2020); Schrittwieser et al. (2020), see two recent reviews in Hamrick et al. (2021); Moerland (2021)).

This surveys focuses on goal-related mechanisms that are mostly orthogonal to the choice of underlying optimization algorithm. In practice, however, most of the research in that space uses DRL methods.

Solving the Multi-Goal RL Problem with Goal-Conditioned RL Algorithms

Goal-conditioned agents see their behavior affected by the goal they pursue. This is formalized via goal-conditioned policies, that is policies which produce actions based on the environment state and the agent's current goal: $\Pi : \mathcal{S} \times \mathcal{Z}_\mathcal{G} \rightarrow \mathcal{A}$, where $\mathcal{Z}_\mathcal{G}$ is the space of goal embeddings corresponding to the goal space $\mathcal{G}$ Schaul et al. (2015a). Note that ensembles of policies can also be formalized this way, via a meta-policy Π that retrieves the particular policy from a one-hot goal embedding z_g Kaelbling (1993); Sutton et al. (2011).

Todo:*[FIX] citation with citet and cite* The idea of using a unique RL agent to target multiple goals dates back to Kaelbling (1993). Later, the HORDE architecture proposed to use interaction experience to update one value function per goal, effectively transferring to all goals the knowledge acquired while aiming at a particular one Sutton et al. (2011). In these approaches, one policy is trained for each of the goals and the data collected by one can be used to train others.

Building on these early results, Schaul et al. (2015a) introduced *Universal Value Function Approximators* (UVFA). They proposed to learn a unique goal-conditioned value function and goal-conditioned policy to replace the set of value functions learned in HORDE. Using neural networks as function approximators, they showed that UVFAs enable transfer between goals and demonstrate strong generalization to new goals.

The idea of *hindsight learning* further improves knowledge transfer between goals Kaelbling (1993); Andrychowicz et al. (2017a). Learning by hindsight, agents can reinterpret a past trajectory collected while pursuing a given goal in the light of a new goal.

By asking themselves, *what is the goal for which this trajectory is optimal?*, they can use the originally failed trajectory as an informative trajectory to learn about another goal, thus making the most out of every trajectory Eysenbach et al. (2020). This ability dramatically increases the sample efficiency of goal-conditioned algorithms and is arguably an important driver of the recent interest in goal-conditioned RL approaches.

2.2.2 Multi-Modal Learning

2.2.3 Compositionality

Part I

Formation of Cultural Models

Chapter 3

Foundations: Emergence of Communication in Population of agents

Contents

Todo:*use O. Sigaud paper to articulate this section* **Todo:***use Jaques as well* **Todo:***use Steel, Cangelosi, and Yoann's background*

3.1 From Experimental Semiotics to Language Games

3.2 From Language Games to Neural Network Agent Communication

Chapter 4

Learning to Guide and to Be Guided in the Architect-Builder Problem

Contents

Abstract

We are interested in interactive agents that learn to coordinate, namely, a *builder* – which performs actions but ignores the goal of the task, i.e. has no access to rewards – and an *architect* which guides the builder towards the goal of the task. We define and explore a formal setting where artificial agents are equipped with mechanisms that allow them to simultaneously learn a task while at the same time evolving a shared communication protocol. Ideally, such learning should only rely on high-level communication priors and be able to handle a large variety of tasks and meanings while deriving communication protocols that can be reused across tasks. The field of Experimental Semiotics has shown the extent of human proficiency at learning from a priori unknown instructions meanings. Therefore, we take inspiration from it and present the Architect-Builder Problem (ABP): an asymmetrical setting in which an architect must learn to guide a builder towards constructing a specific structure. The architect knows the target structure but cannot act in the environment and can only send arbitrary messages to the builder. The builder on the other hand can act in the environment, but receives no rewards nor has any knowledge about the task, and must learn to solve it relying only on the messages sent by the architect. Crucially, the meaning of messages is initially not defined nor shared between the agents but must be negotiated throughout learning. Under these constraints, we propose Architect-Builder Iterated Guiding (ABIG), a solution

to the Architect-Builder Problem where the architect leverages a learned model of the builder to guide it while the builder uses self-imitation learning to reinforce its guided behavior. To palliate to the non-stationarity induced by the two agents concurrently learning, ABIG structures the sequence of interactions between the agents into interaction frames. We analyze the key learning mechanisms of ABIG and test it in a 2-dimensional instantiation of the ABP where tasks involve grasping cubes, placing them at a given location, or building various shapes. In this environment, ABIG results in a low-level, high-frequency, guiding communication protocol that not only enables an architect-builder pair to solve the task at hand, but that can also generalize to unseen tasks.

4.1 Motivations

Humans are notoriously successful at teaching – and learning from – each others. This enables skills and knowledge to be shared and passed along generations, being progressively refined towards mankind's current state of proficiency. People can teach and be taught in situations where there is no shared language and very little common ground, such as a parent teaching a baby how to stack blocks during play. Experimental Semiotics Galantucci & Garrod (2011), a line of work that studies the forms of communication that people develop when they cannot use pre-established ones, reveals that humans can even teach and learn without direct reinforcement signal, demonstrations or a shared communication protocol. Vollmer et al. (2014) for example investigate a co-construction (CoCo) game experiment where an architect must rely only on arbitrary instructions to guide a builder toward constructing a structure. In this experiment, both the task of building the structure and the meanings of the instructions – through which the architect guides the builder – are simultaneously learned throughout interactions. Such flexible teaching – and learning – capabilities are essential to autonomous artificial agents if they are to master an increasing number of skills without extensive human supervision. As a first step toward this research direction, we draw inspiration from the CoCo game and propose the *Architect-Builder Problem* (ABP): an interactive learning setting that models agents' interactions with *Markov Decision Processes* Puterman (2014) (MDPs). In the ABP learning has to occur in a social context through observations and communication, in the absence of direct imitation or reinforcement Bandura & Walters (1977). Specifically, the constraints of the ABP are: (1) the builder has absolutely no knowledge about the task at hand (no reward and no prior on the set of possible tasks), (2) the architect can only interact with the builder through communication signals (cannot interact with the environment or provide demonstrations), and (3) the communication signals have no pre-defined meanings (nor belong to a set of known possible meanings). (1) sets this work apart from Reinforcement Learning (RL) and even Multi-Agent RL (MARL) where explicit rewards are available to all agents. (2) implies the absence of tele-operation or third-person demonstrations and thus distinguishes the ABP from Imitation and Inverse Reinforcement Learning (IRL). Finally, (3) prevents the architect from relying on a fixed communication protocol since the meanings of instructions must be negotiated.

These constraints make ABP an appealing setting to investigate *Human-Robot Interaction* (HRI) Goodrich & Schultz (2008) problems where "a learner tries to figure out what a teacher wants them to do" Grizou et al. (2013); Cederborg & Oudeyer (2014).

Specifically, the challenge of *Brain Computer Interfaces* (BCI), where users use brain signals to control virtual and robotic agents in sequential tasks Katyal et al. (2014); deBettencourt et al. (2015); Mishra & Gazzaley (2015); Muñoz-Moldes & Cleeremans (2020); Chiang et al. (2021), is well captured by the ABP. In BCIs, (3) is identified as the calibration problem and is usually tackled with supervised learning to learn a mapping between signals and meanings. As this calibration phase is often laborious and impractical for users, current approaches investigate calibration-free solutions where the mapping is learned interactively Grizou et al. (2014); Xie et al. (2021). Yet, these works consider that the user (i.e. the architect) is fixed, in the sense that it does not adapt to the agent (i.e. the builder) and uses a set of pre-defined instructions (or feedback) meanings that the agent must learn to map to signals. In our ABP formulation however, the architect is dynamic and, as interactions unfold, must learn to best guide a learning builder by tuning the meanings of instructions according to the builder's reactions. In that sense, ABP provides a more complete computational model of agent-agent or human-agent interaction.

With all these constraints in mind, we propose Architect Builder Iterated Guiding (ABIG), an algorithmic solution to ABP when both agents are AIs. ABIG is inspired by the field of experimental semiotics and relies on two high-level interaction priors: *shared intent* and *interaction frames*. Shared intent refers to the fact that, although the builder ignores the objective of the task to fulfill, it will assume that its objective is aligned with the architect's. This assumption is characteristic of cooperative tasks and shown to be a necessary condition for the emergence of communication both in practice Foerster et al. (2016); Cao et al. (2018) and in theory Crawford & Sobel (1982). Specifically, the builder should assume that the architect is guiding it towards a shared objective. Knowing this, the builder must reinforce the behavior it displays when guided by the architect. We show that the builder can efficiently implement this by using imitation learning on its own guided behavior. Because the builder imitates itself, we call it self-imitation. The notion of *interaction frames* (also called *pragmatic frames*) states that agents that interact in sequence can more easily interpret the interaction history Bruner (1985); Vollmer et al. (2016). In ABIG, we consider two distinct interaction frames. These are stationary which means that when one agent learns, the other agent's behavior is fixed. During the first frame (the modelling frame), the builder is fixed and the architect learns a model of the builder's message-conditioned behavior. During the second frame (the guiding frame), the architect is fixed and the builder learns to be guided via self-imitation learning.

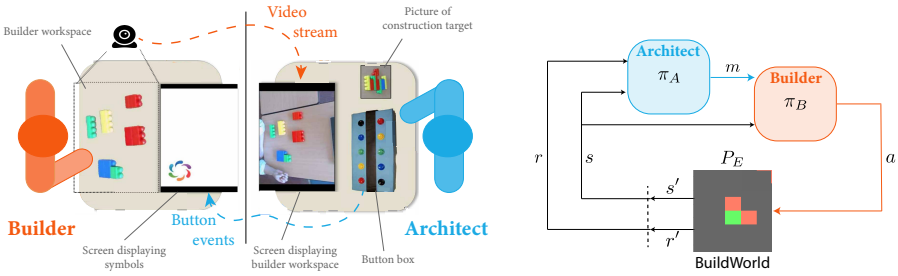
We show that ABIG results in a low-level, high-frequency, guiding communication protocol that not only enables an architect-builder pair to solve the task at hand, but can also be used to solve unseen tasks. **Our contributions are:**

- The Architect-Builder Problem (ABP), an interactive learning setting to study how artificial agents can simultaneously learn to solve a task and derive a communication protocol.
- Architect-Builder Iterated Guiding (ABIG), an algorithmic solution to the ABP.
- An analysis of ABIG's key learning mechanisms.
- An evaluation of ABIG on a construction environment where we show that ABIG agents evolve communication protocols that generalize to unseen harder tasks.

- A detailed analysis of ABIG's learning dynamics and impact on the mutual information between messages and actions (in the Supplementary Material).

4.2 The Architect-Builder Problem

The Architect-Builder Problem. We consider a multi-agent setup composed of two agents: an architect and a builder. Both agents observe the environment state s but only the architect knows the goal at hand. The architect cannot take actions in the environment but receives the environmental reward r whereas the builder does not receive any reward and has thus no knowledge about the task at hand. In this asymmetrical setup, the architect can only interact with the builder through a communication signal m sampled from its policy $\pi_A(m|s)$. These messages, that have no a priori meanings, are received by the builder which acts according to its policy $\pi_B(a|s,m)$. This makes the environment transition to a new state s' sampled from $P_E(s'|s,a)$ and the architect receives reward r'. Messages are sent at every time-step. The CoCo game that inspired ABP is sketched in Figure 4.1(a) while the overall architect-builder-environment interaction diagram is given in Figure 4.1(b). The differences between the ABP setting and the MARL and IRL settings are illustrated in Figure A.2.



(a) (b)

Figure 4.1: (a) **Schematic view of the CoCo Game (the inspiration for ABP).** The architect and the builder should collaborate in order to build the construction target while located in different rooms. The architecture has a picture of the target while the builder has access to the blocks. The architect monitors the builder workspace via a camera (video stream) and can communicate with the builder only through the use of 10 symbols (button events). (b) **Interaction diagram between the agents and the environment in our proposed ABP.** The architect communicates messages (m) to the builder. Only the builder can act (a) in the environment. The builder conditions its action on the message sent by the builder ($\pi_B(a|s,m)$). The builder never perceives any reward from the environment. A schematic view of the equivalent ABP problem is provided in Figure A.1(b).

BuildWorld. We conduct our experiments in *BuildWorld*. BuildWorld is a 2D construction grid-world of size $(w \times h)$. At the beginning of an episode, the agent and N_b blocks are spawned at different random locations. The agent can navigate in this world and grasp blocks by activating its gripper while on a block. The action space $\mathcal{A}$ is discrete and include a "do nothing" action ($|\mathcal{A}| = 6$). At each time step, the agent observes its position in the grid, its gripper state as well as the position of all the blocks and if they are grasped ($|\mathcal{S}| = 3 + 3N_b$).

Tasks. BuildWorld contains 4 different training tasks: 1) 'Grasp': The agent must grasp any of the blocks; 2) 'Place': The agent must place any block at a specified location in the grid; 3/4) 'H-Line/V-line': The agent must place all the blocks in a horizontal/vertical line configuration. BuildWorld also has a harder fifth testing task, '6-blocks-shapes', that consists of more complex configurations and that is used to challenge an algorithm's transfer abilities. For all tasks, rewards are sparse and only given when the task is completed.

This environment encapsulates the interactive learning challenge of ABP while removing the need for complex perception or locomotion. In the RL setting, where the same agent acts and receives rewards, this environment would not be very impressive. However, it remains to be shown that the tasks can be solved in the setting of ABP (with a reward-less builder and an action-less architect).

Communication. The architect guides the builder by sending messages m which are one-hot vectors of size $|\mathcal{V}|$ ranging from 2 to 72, see Suppl. Section A.2.3 for the impact of this parameter.

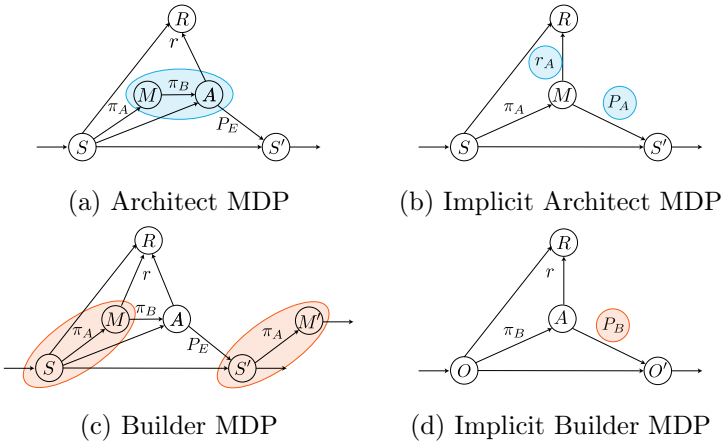
Additional Assumptions. In order to focus on the architect-builder interactions and the learning of a shared communication protocol, the architect has access to $P_{E}(s'|s,a)$ and to the reward function $r(s,a)$ of the goal at hand. This assumes that, if the architect were to act in the environment instead of the builder, it would be able to quickly figure out how to solve the task. This assumption is compatible with the CoCo game experiment Vollmer et al. (2014) where humans participants, and in particular the architects, are known to have such world models.

4.3 ABIG: Architect-Builder Iterated Guiding

4.3.1 Analytical description

Agents-MDPs. In the Architect-Builder Problem, agents are operating in different, yet coupled, MDPs. Those MDPs depend on their respective point of view (see Figure 4.2). From the point of view of the architect, messages are actions that influence the next state as well as the reward (see Figure 4.2 (a)). The architect knows the environment transition function $P_{E}(s'|s,a)$ and $r(s,a)$, the true reward function associated with the task that does not depend explicitly on messages. It can thus derive the effect of its messages on the builder's actions that drive the reward and the next states (see Figure 4.2 (b)). On the other hand, the builder's state is composed of the environment state and the message, which makes estimating state transitions challenging as one must also capture the message dynamics (see Figure 4.2 (c)). Yet, the builder can leverage its knowledge of the architect picking messages based on the current environment state. The equivalent transition and reward models, when available, are given below (see derivations in Suppl. Section A.1).

$$\left.\begin{aligned} P_{A}(s'|s,m) &= \sum_{a\in\mathcal{A}} \tilde{\pi}_{B}(a|s,m) P_{E}(s'|a,s) \\ r_{A}(s,m) &= \sum_{a\in\mathcal{A}} \tilde{\pi}_{B}(a|s,m) r(s,a) \end{aligned}\right\} \quad \text{with} \quad \tilde{\pi}_{B}(a|s,m) \triangleq P(a|s,m) \tag{4.1}$$



(a) Architect MDP (b) Implicit Architect MDP

(c) Builder MDP (d) Implicit Builder MDP

Figure 4.2: **Agent's Markov Decision Processes.** Highlighted regions refer to MDP coupling. (a) The architect's transitions and rewards are conditioned by the builder's policy π_B. (b) Architect's MDP where transition and reward models implicitly account for builder's behavior. (c-d) The builder's transition model depends on the architect's message policy π_A. The builder's learning signal r is unknown.

$$P_B(s', m'|s, m, a) = \tilde{\pi}_A(m'|s')P_E(s'|s, a) \quad \text{with} \quad \tilde{\pi}_A(m'|s') \triangleq P(m'|s') \tag{4.2}$$

where subscripts A and B refer to the architect and the builder, respectively. $\tilde{x}$ denotes that x is unknown and must be approximated. From the builder's point of view, the reward – denoted $\tilde{r}$ – is unknown. This prevents the use of classical RL algorithms.

Shared Intent and Interaction Frames. It follows from Eq. (4.1) that, provided that it can approximate the builder's behavior, the architect can compute the reward and transition models of its MDP. It can then use these to derive an optimal message policy π_A^* that would maximize its objective:

$$\pi_A^* = \underset{\pi_A}{\text{argmax}}\, G_A = \underset{\pi_A}{\text{argmax}}\, \mathbb{E}[\sum_t \gamma^t r_{A,t}] \tag{4.3}$$

$\gamma \in [0,1]$ is a discount factor and the expectation can be thought of in terms of π_A, P_A and the initial state distribution. However, the expectation can also be though in terms of the corresponding trajectories $\tau \triangleq \{(s, m, a, r)_t\}$ generated by the architect-builder interactions. In other words, when using π_A^* to guide the builder, the architect-builder pair generates trajectories that maximizes G_A. The builder has no reward signal to maximize, yet, it relies on a shared intent prior and assumes that its objective is the same as the architect's one:

$$G_B = G_A = \mathbb{E}_\tau[\sum_t \gamma^t r_{A,t}] = \mathbb{E}_\tau[\sum_t \gamma^t \tilde{r}_t] \tag{4.4}$$

where the expectations are taken with respect to trajectories τ of architect-builder interactions. Therefore, under the shared intent prior, architect-builder interactions where the architect uses π_A^* to maximize G_A also maximize G_B. This means that the builder can interpret these interaction trajectories as demonstrations that maximize its unknown reward function $\tilde{r}$. Consequently, the builder can reinforce the desired behavior – towards

which the architect guides it – by performing self-Imitation Learning[1] on the interaction trajectories τ.

Note that in Eq. (4.1), the architect's models can be interpreted as expectations with respect to the builder's behavior. Similarly, the builder's objective depends on the architect's guiding behavior. This makes one agent's MDP highly non-stationary and the agent must adapts its behavior if the other agent's policy changes. To palliate to this, agents rely on interaction frames which means that, when one agent learns, the other agent's policy is fixed to restore stationarity. The equivalent MDPs for the architect and the builder are respectively $\mathcal{M}_A = \langle \mathcal{S}, \mathcal{V}, P_A, r_A, \gamma \rangle$ and $\mathcal{M}_B = \langle \mathcal{S} \times \mathcal{V}, \mathcal{A}, P_B, \emptyset, \gamma \rangle$. Finally, $\pi_A : \mathcal{S} \mapsto \mathcal{V}$, $P_A : \mathcal{S} \times \mathcal{V} \mapsto [0, 1]$, $r_A : \mathcal{S} \times \mathcal{V} \mapsto [0, 1]$, $\pi_B : \mathcal{S} \times \mathcal{V} \mapsto \mathcal{A}$ and $P_B : \mathcal{S} \times \mathcal{V} \times \mathcal{A} \mapsto [0, 1]$ where $\mathcal{S}, \mathcal{A}$ and $\mathcal{V}$ are respectively the sets of states, actions and messages.

4.3.2 Practical Algorithm

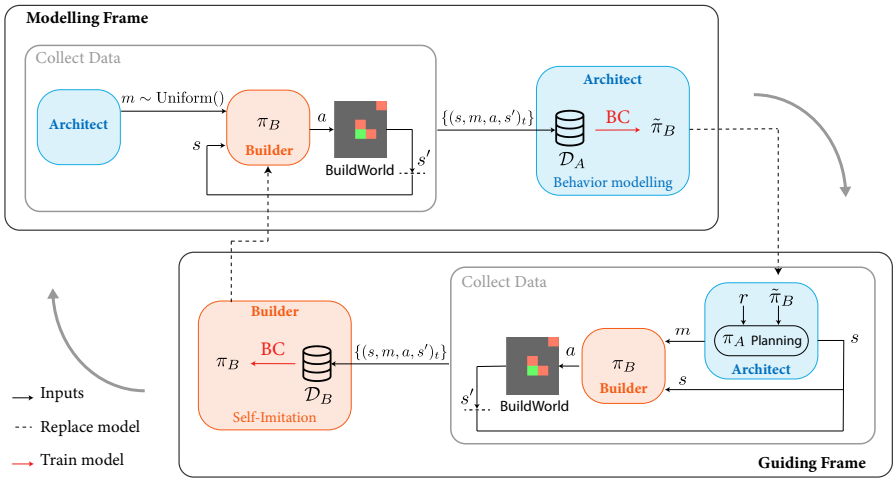


Figure 4.3: **Architect-Builder Iterated Guiding.** Agents iteratively interact through the modelling and guiding frames. In each frame, one agent collects data and improves its policy while the other agent's behavior is fixed.

ABIG iteratively structures the interactions between a builder-architect pair into interaction frames. Each iteration starts with a *modelling frame* during which the architect learns a model of the builder. Directly after, during the *guiding frame*, the architect leverages this model to produce messages that guide the builder. On its side, the builder stores the guiding interactions to train and refine its policy π_B. The interaction frames are described below. The algorithm is illustrated in Figure 4.3 and the pseudo-code is reported in **Algorithm 2 in Suppl. Section A.1.3**.

Modelling Frame. The architect records a data-set of interactions $\mathcal{D}_A \triangleq \{(s, m, a, s')_t\}$ by sending random messages m to the builder and observing its reaction. After collect-

[1] not to be confused with Oh et al. (2018) which is an off-policy actor-critic algorithm promoting exploration in single-agent RL.

ing enough interactions, the architect learns a model of the builder $\tilde{\pi}_{B}$ using *Behavioral Cloning* (BC) Pomerleau (1991).

Guiding Frame. During the guiding frame, the architect observes the environment states s and produces messages so as to maximize its return (see Eq. 4.3). The policy of the architect is a Monte Carlo Tree Search Algorithm (MCTS) Kocsis & Szepesvári (2006) that searches for the best message by simulating the reaction of the builder using $\tilde{a} \sim \tilde{\pi}_{B}(\cdot|m, s)$ alongside the dynamics and reward models. During this frame, the builder stores the interactions in a buffer $\mathcal{D}_{B} \triangleq \{(s, m, a, s')_t\}$. At the end of the guiding frame, the builder self-imitates by updating its policy π_{B} with BC on $\mathcal{D}_{B}$.

Practical Considerations. All models are parametrized by two-hidden layer 126-units feedforward ReLu networks. BC minimizes the cross-entropy loss with Adam optimizer Kingma & Ba (2015). Networks are re-initialized before each BC training. The architect's MCTS uses Upper-Confidence bound for Trees and relies on heuristics rather than Monte-Carlo rollouts to estimate the value of states. For more details about training, MCTS and hyper-parameters please see Suppl. Section A.1.3.

The resulting method (ABIG) is general and can handle a variety of tasks while not restricting the kind of communication protocol that can emerge. Indeed, it only relies on a few high-level priors, namely, the architect's access to environment models, shared intent and interaction frames.

In addition to ABIG we also investigate two control settings: ABIG *-no-intent* – the builder interacts with an architect that disregards the goal and therefore sends random messages during training. At evaluation, the architect has access to the exact model of the builder ($\tilde{\pi}_{B} = \pi_{B}$) and leverages it to guide it towards the evaluation goal (the architect no longer disregards the goal). And *random* – the builder takes random actions. The comparison between ABIG and ABIG-no-intent measures the impact of doing self-imitation on guiding versus on non-guiding trajectories. The random baseline is used to provide a performance lower bound that indicates the task's difficulty.

4.3.3 Understanding the Learning Dynamics

Architect-Builder Iterated Guiding relies on two steps. First, the architect selects *favorable* messages, i.e. messages that maximize the likelihood of the builder picking optimal actions with respect to the architect's reward. Then, the builder does self-imitation and reinforces the guided behavior by maximizing the likelihood of the corresponding messages-actions sequence under its policy. The message-to-action associations (or preferences) are encoded in the builder's policy $\pi_{B}(a|s, m)$. Maximum likelihood assumes that actions are initially equiprobable for a given message. Therefore, actions under a message that is not present in the data-set ($\mathcal{D}_{B}$) remains so. In other words, if the builder never observes a message, it assumes that this message is equally associated with all the possible actions. This enables the builder to *forget* past message-to-action associations that are not used – and thus not reinforced – by the architect. In practice, initial uniform likelihood is ensured by resetting the builder's policy network before each self-imitation. The architect can leverage the forget mechanism to erase unfavorable associations until a favorable one emerges. Such favorable associations can then be reinforced by the

architect-builder pair until it is made deterministic. The *reinforcement* process of favorable associations is also enabled by the self-imitation phase. Indeed, for a given message m, the self-imitation objective for π on a data-set $\mathcal{D}$ collected using π is:

$$J(m,\pi) = -\sum_{a\sim\mathcal{D}} \log \pi(a|m) \approx \mathbb{E}_{a\sim\pi(\cdot|m)}[-\log \pi(a|m)] \approx H[\pi(\cdot|m)] \tag{4.5}$$

where H stands for the entropy of a distribution. Therefore, maximizing the likelihood in this case results in minimizing the entropy of $\pi(\cdot|m)$ and thus reinforces the associations between messages and actions. Using these mechanisms the architect can adjust the policy of the builder until it becomes *controllable*, i.e. deterministic (strong preferences over actions for a given message) and flexible (varied preferences across messages). Conversely, in the case of ABIG-no-intent, the architect does not guide the builder and simply sends messages at random. Favorable and unfavorable messages are thus sampled alike which prevents the forget mechanism to undo unfavorable message-to-action associations. Consequently in that case, self-imitation tends to simply reinforce initial builder's preferences over actions making the controllability of the builder policy depend heavily on the initial preferences. We illustrate the above learning mechanisms in Suppl. Section A.1.4 by applying ABIG to a simple instantiation of the ABP. Figure A.3 and Figure A.5 confirm that ABIG uses the forget and reinforcement mechanisms to circumvent the unfavorable initial conditions while ABIG-no-intent simply reinforces them. Eventually, Figure A.5 reports that ABIG always reaches 100% success rate regardless of the initial conditions while ABIG-no-intent success rate depends on the initial preferences (only 3% when they are unfavorable).

Interestingly, the emergent learning mechanisms discussed here are reminiscent of the amplification and self-enforcement of random fluctuations in naming games Steels (1995a). In naming games however, the self-organisation of vocabularies are driven by each agent maximizing its communicative success whereas in our case the builder has no external learning signal and simply self-imitates.

4.3.4 Related Work

This work is inspired by experimental semiotics Galantucci & Garrod (2011) and in particular Vollmer et al. (2014) that studied the CoCo game with human subjects as a key step towards understanding the underlying mechanisms of the emergence of communication. Here we take a complementary approach by defining and investigating solutions to the ABP, a general formulation of the CoCo game where both agents are AIs.

Recent MARL work Lowe et al. (2017); Woodward et al. (2020); Roy et al. (2020); Ndousse et al. (2021), investigate how RL agents trained in the presence of other agents leverage the behaviors they observe to improve learning. In these settings, the other agents are used to build useful representation or gain information but the main learning signal of every agent remains a ground truth reward.

Feudal Learning Dayan & Hinton (1992); Kulkarni et al. (2016); Vezhnevets et al. (2017); Nachum et al. (2018); Ahilan & Dayan (2019) investigate a setting where a manager sets the rewards of workers to maximize its own return. In this Hierarchical setting, the manager interacts by directly tweaking the workers' learning signal. This

would be unfeasible for physically distinct agents, hence those methods are restricted to single-agent learning. On the other hand, ABP considers separate agents, that must hence communicate by influencing each other's observations instead of rewards signals.

Inverse Reinforcement Learning (IRL) Ng et al. (2000) and Imitation Learning (IL) Pomerleau (1991) have been investigated for HRI when it is challenging to specify a reward function. Instead of defining rewards, IRL and IL rely on expert demonstrations. Hadfield-Menell et al. (2016) argue that learning from expert demonstrations is not always optimal and investigate how to produce instructive demonstrations to best teach an apprentice. Crucially, the expert is aware of the mechanisms by which the apprentice learns, namely RL on top of IRL. This allows the expert to assess how its demonstrations influence the apprentice policy, effectively reducing the problem to a single agent POMDP. In our case however, the architect and the builder do not share the same action space which prevents the architect from producing demonstrations. In addition, the architect ignores the builder's learning process which makes the simplification to a single agent teacher problem impossible.

In essence, the ABP is closest to works tackling the calibration-free BCI control problem Grizou et al. (2014); Xie et al. (2021). Yet, these works both consider that the architect sends messages after the builder's actions and thus enforce that the feedback conveys a reward. Crucially, the architect does not learn and communicates with a fixed mapping between feedback and pre-defined meanings ("correct" vs. "wrong"). Those meanings are known to the builder and it simply has to learn the mapping between feedback and meaning. In our case however, the architect communicates before the builder's action and thus rather gives instructions than feedback. Additionally, the builder has no a priori knowledge of the set of possible meanings and the architect adapts those to the builder's reaction. Finally, Grizou et al. (2013) handles both feedback and instruction communications but relies on known task distribution and set of possible meanings. In terms of motivations, previous works are interested in one robot figuring out a fixed communication protocol while we train two agents to collectively emerge one.

Our BuildWorld resembles GridLU proposed by Bahdanau et al. (2019b) to analyze reward modelling in language-conditioned learning. However, their setting is fundamentally different to ours as it investigates single agent goal-conditioned IL where goals are predefined episodic linguistic instructions labelling expert demonstrations. Nguyen et al. (2021) alleviate the need for expert demonstrations by introducing an interactive teacher that provides descriptions of the learning agent's trajectories. In this HRI setting, the teacher still follows a fixed pre-defined communication protocol known by the learner: messages are activity descriptions. Our ABP formulation relates to the Minecraft Collaborative Building Task Narayan-Chen et al. (2019) and the IGLU competition Kiseleva et al. (2021); however, they do not consider emergent communication. Rather, they focus on generating architect utterances by leveraging a human-human dialogues corpus to learn pre-established meanings expressed in natural language. Conversely, in ABP both agents learn and must evolve the meanings of messages while solving the task without relying on any form of demonstration.

4.4 Experiments

In the following sections, success rates (sometimes referred as scores) are averaged over 10 random seeds and error bars are ±2SEM with SEM the Standard Error of the Mean. If not stated otherwise, the grid size is (5×6), contains three blocks ($N_b = 3$) and the vocabulary size is $|\mathcal{V}| = 18$.

ABIG's learning performances. We apply ABIG to the four learning tasks of BuildWorld and compare it with the two control settings: ABIG-no-intent (no guiding during training) and random (builder takes random actions). Figure 4.4 reports the mean success rate on the four tasks defined in Section 4.2. First, we observe that ABIG significantly outperforms the control conditions on all tasks. Second, we notice that on the simpler 'grasp' task ABIG-no-intent achieves a satisfactory mean score of 0.77±0.03. This is consistent with the learning dynamic analysis provided in Suppl. Section A.1.4 that shows that, in favorable settings, a self-imitating builder can develop a reasonably controllable policy (defined in Section 4.3.3) even if it learns on non-guiding trajectories. Nevertheless, when the tasks get more complicated and involve placing objects or drawing lines, the performances of ABIG-no-intent drop significantly whereas ABIG continues to achieve high success rates (> 0.8). This demonstrates that ABIG enables a builder-architect pair to successfully agree on a communication protocol that makes the builder's policy controllable and enables the architect to efficiently guide it.

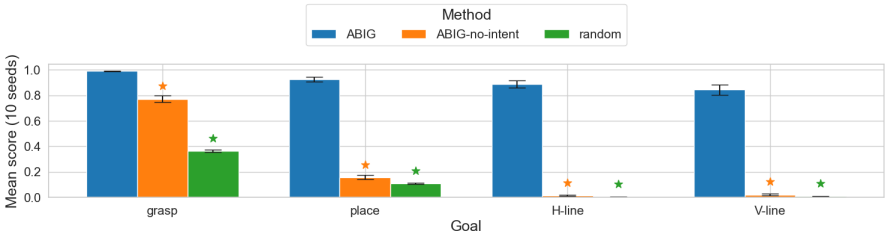


Figure 4.4: Methods performances (stars indicate significance with respect to ABIG model according to Welch's t-test with null hypothesis $\mu_1 = \mu_2$, at level $\alpha = 0.05$). ABIG outperforms control baselines on all goals.

ABIG's transfer performances.

Building upon previous results, we propose to study whether a learned communication protocol can transfer to new tasks. The architect-builder pairs are trained on a single task and then evaluated without retraining on the four tasks. In addition, we include 'all-goals': a control setting in which the builder learns a single policy by being guided on all four goals during training. Figure 4.5 shows that, on all training tasks except 'grasp', ABIG enables a transfer performance above 0.65 on all testing tasks. Notably, training on 'place' results in a robust communication protocol that can be used to solve the other tasks with a success rate above 0.85, being effectively equivalent as training on 'all-goals' directly. This might be explained by the fact that placing blocks at specified locations is an atomic operation required to build lines.

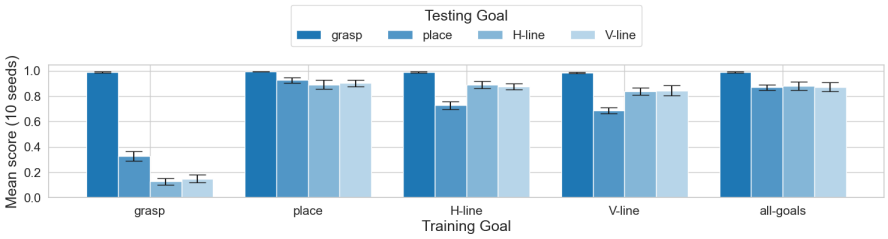


Figure 4.5: ABIG transfer performances without retraining depending on the training goal. ABIG agents learn a communication protocol that transfers to new tasks. Highest performances reached when training on 'place'.

Challenging ABIG's transfer abilities. Motivated by ABIG's transfer performances, we propose to train it on the 'place' task in a bigger grid (6×6) with $N_b = 6$ and $|\mathcal{V}| = 72$. Then, without retraining, we evaluate it on the '6-block-shapes' task[2] that consists in constructing the shapes given in Figure 4.6. The training performance on 'place' is 0.96 ± 0.02 and the transfer performance on the '6-block-shapes' is 0.85 ± 0.03. This further demonstrates ABIG's ability to derive robust communication protocols that can solve more challenging unseen tasks. **Additional experiments.** The Supplementary Materials contains the following experiments:

Figure 4.6: 6-block-shapes that ABIG can construct in transfer mode when trained on the 'place' task.

- Figures A.3, A.4 and A.5 analyse the builder's message-to-action preferences. They illustrate ABIG's learning mechanisms (forget and reinforce) and compare them to ABIG-no-intent's.
- Figure A.6 shows that, as the communication protocol settles, the message-action mutual information becomes greater than the state-action mutual information which is a desirable feature for the emergence of communication.
- Figure A.7 reports ABIG outperforming complementary baselines.
- Figure A.8 shows ABIG's performance increasing with the vocabulary size, suggesting that with more messages available, the architect can more efficiently refer to the desired action.

4.5 Discussion and future work

This work formalizes the ABP as an interactive setting where learning must occur without explicit reinforcement, demonstrations or a shared language. To tackle ABP, we propose ABIG: an algorithm allowing to learn how to guide and to be guided. ABIG is based only

[2]For rollouts see https://sites.google.com/view/architect-builder-problem/

on two high-level priors to communication emergence (shared intent and interactions frames). ABP's general formulation allows us to formally enforce those priors during learning. We study their influence through ablation studies, highlighting the importance of shared intent achieved by doing self-imitation on guiding trajectories. When performed in interaction frames, this mechanism enables agents to evolve a communication protocol that allows them to solve all the tasks defined in BuildWorld. More impressively, we find that communication protocols derived on a simple task can be used to solve harder, never-seen goals.

Our approach has several limitations which open up different opportunities for further work. First, ABIG trains agents in a stationary configuration which implies doing several interaction frames. Each interaction frame involves collecting numerous transitions. Thus, ABIG is not data efficient. A challenging avenue would be to relax this stationarity constraint and have agents learn from buffers containing non-stationary data with obsolete agent behaviors. Second, the builder remains dependent on the architect's messages even at convergence. Using a Vygotskian approach Colas et al. (2020b, 2021b), the builder could internalize the guidance from the architect to become autonomous in the task. This could, for instance, be achieved by having the builder learn a model of the architect's message policy once the communication protocol has converged.

Because we present the first step towards interactive agents that learn in the ABP, our method uses simple tools (feed-forward networks and self-imitation learning). It is however important to note that our proposed formulation of the ABP can support many different research directions. Experimenting with agents' models could allow for the investigation of other forms of communication. One could, for instance, include memory mechanisms in the models of agents in order to facilitate the emergence of retrospective feedback, a form of emergent communication observed in Vollmer et al. (2014). ABP is also compatible with low-frequency feedback. As a further experiment in this direction, one could penalize the architect for sending messages and assess whether a pair can converge to higher-level meanings. Messages could also be composed of several tokens in order to allow for the emergence of compositionality. Finally, our proposed framework can serve as a testbed to study the fundamental mechanisms of emergent communication by investigating the impact of high level communication priors from experimental semiotics.

Chapter 5

Emergence of Graphical language

Contents

Abstract

The framework of Language Games studies the emergence of languages in populations of agents. Recent contributions relying on deep learning methods focused on agents communicating via an idealized communication channel, where utterances produced by a speaker are directly perceived by a listener. This comes in contrast with human communication, which instead relies on a *sensory-motor channel*, where motor commands produced by the speaker (e.g. vocal or gestural articulators) result in sensory effects perceived by the listener (e.g. audio or visual). Here, we investigate if agents can evolve a shared language when they are equipped with a continuous sensory-motor system to produce and perceive signs, e.g. drawings. To this end, we introduce the Graphical Referential Game (GREG) where a speaker must produce a graphical utterance to name a visual referent object consisting of combinations of MNIST digits while a listener has to select the corresponding object among distractor referents, given the produced message. The utterances are drawing images produced using dynamical motor primitives combined with a sketching library. To tackle GREG we present CURVES: a multimodal contrastive deep learning mechanism that represents the energy (alignment) between named referents and utterances generated through gradient ascent on the learned energy landscape. We, then, present a set of experiments and metrics based on a systematic compositional dataset to evaluate the resulting language. We show that our method allows the emergence of a shared, graphical language with compositional properties.

5.1 Motivations

Understanding the emergence and evolution of human languages is a significant challenge that has involved many fields, from linguistics to developmental cognitive sciences (Chris-

tiansen & Kirby, 2003). Computational experimental semiotics (Galantucci & Garrod, 2011) has seen some success in modeling the formation of communication systems in populations of artificial agents (Cangelosi & Parisi, 2002; Kirby et al., 2014). More specifically, *Language Game* models, such as naming games (Steels & Loetzsch, 2012), have been used to show how a population of agents can self-organize a culturally shared lexicon without centralized coordination. Given the recent successes of artificial neural networks in solving complex tasks such as image classification (Krizhevsky et al., 2012; He et al., 2015b, 2016; Dosovitskiy et al., 2021) and natural language understanding (Devlin et al., 2019; Radford et al., 2019; Brown et al., 2020), many works have leveraged them to study the emergence of communication in groups of agents (Lazaridou & Baroni, 2020), mainly using multi-agent deep reinforcement learning and language games (Nguyen et al., 2020; Mordatch & Abbeel, 2018; Lazaridou et al., 2018; Portelance et al., 2021; Chaabouni et al., 2021). These advances have made it possible to scale up language game models to environments where linguistic conventions are jointly learned with visual representations of raw image perception, as well as to environments where emergent communication is used as a tool to achieve joint cooperative tasks (Barde et al., 2022).

So far, most of these methods have considered only idealized symbolic communication channels based on discrete tokens (Lazaridou et al., 2017; Mordatch & Abbeel, 2018; Chaabouni et al., 2021) or fixed-size sequences of word tokens (Havrylov & Titov, 2017; Portelance et al., 2021). This predefined means of communication is motivated by language's discrete and compositional nature. But how can this specific structure emerge during vocalization or drawing, for instance? Although fundamental in the investigation of the origin of language (Dessalles, 2000; Cheney & Seyfarth, 2005; Oller et al., 2019), this question seems to be neglected by recent approaches to Language Games (Moulin-Frier & Oudeyer, 2020). We, therefore, propose to study how communication could emerge between agents producing and perceiving continuous signals with a constrained *sensory-motor system*.

Such continuous constrained systems have been used in the cognitive science literature as models of sign production to study the self-organization of speech in artificial systems (de Boer, 2000; Oudeyer, 2006; Moulin-Frier et al., 2015). In this paper, we focus on a drawing sensory-motor system producing graphical signs. The sensory-motor system is made of Dynamical Motor Primitives (DMPs) (Schaal, 2006) combined with a sketching system (Mihai & Hare, 2021a) enabling the conversion of motor commands into images. Drawing systems have the advantage of producing 2D trajectories interpretable by humans while preserving the non-linear properties of speech models, which were shown to ease the discretization of the produced signals (Stevens, 1989; Moulin-Frier et al., 2015). We introduce the *Graphical Referential Game*: a variation of the original referential game, where a *Speaker* agent (top of Figure 5.1) has to produce a graphical *utterance* given a single target *referent* while a *Listener* agent (bottom of Figure 5.1) has to select an element among a context made of several referents, given the produced utterance (agents alternate their roles). In this setting, we first investigate whether a population of agents can converge on an efficient communication protocol to solve the graphical language game. Then, we evaluate the coherence and compositional properties of the emergent language, since it is one of the main characteristics of human languages.

Early language game implementations (Steels, 1995b, 2001) achieve communication convergence by using contrastive methods to update association tables between object

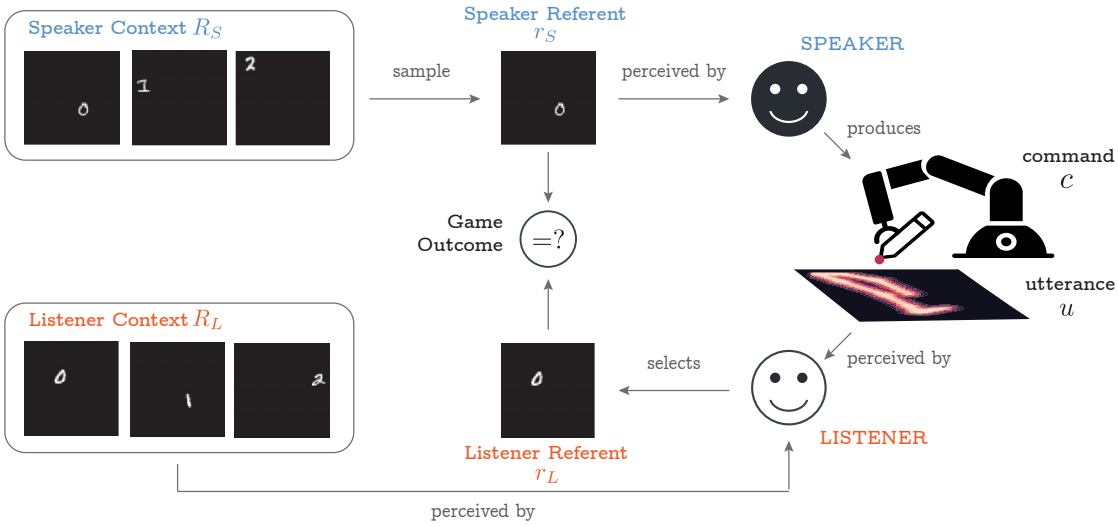


Figure 5.1: **The Graphical Referential Game:** During an instantiation of the game, the speaker's goal is to produce a motor command c that will yield an utterance u in order to denote a referent r_S sampled from a context $\tilde{R}_S$. Following this step, the listener needs to interpret the utterance in order to guess the referent it denotes among a context $\tilde{R}_L$. The game is a success if the listener and the speaker agree on the referent ($r_L \equiv r_S$).

referents and utterances. While recent works use deep learning methods to target high-dimensional signals they do not explore contrastive approaches. Instead, they model interactions as a multi-agent reinforcement learning problem where utterances are actions, and agents are optimized with policy gradients, using the outcomes of the games as the reward signal (Lazaridou et al., 2017). In the meantime, recent models leveraging contrastive multimodal mechanisms such as CLIP (Radford et al., 2021) have achieved impressive results in modeling associations between images and texts. Combined with efficient generative methods (**?**), they can compose textual elements that are reflected in image form as the composition of their associated visual concepts. Inspired by these techniques, we propose CURVES: Contrastive Utterance-Referent associatiVE Scoring, an algorithmic solution to the graphical referential game. CURVES relies on two mechanisms: 1) The contrastive learning of an energy landscape representing the alignment between utterances and referents and 2) the generation of utterances that maximize the energy for a given target referent. We evaluate CURVES in two instantiations of the graphical referential game: one with symbolic referents encoded by one-hot vectors and another with visual referents derived from the multiple MNIST digits (LeCun et al., 1998). We show that CURVES converges to a shared graphical language that enables a population of agents not only to name complex visual referents but also to name new referent compositions that were never encountered during training.

Scope.

The idea of using a sensory-motor system to study the emergence of forms of combinatoriality in language dates back to methods investigating the origins of digital vocalization systems (de Boer, 2000; Oudeyer, 2005; Zuidema & De Boer, 2009). Such studies were conducted in the context of imitation games at the level of phonemes to observe

the formation of speech utterances (syllables, words) that were systematically composed from lower-level meaningless elements (phonemes). This corresponded to the first level of compositionality within the notion of duality of patterning (Hockett & Hockett, 1960). Yet, these works did not consider referential games and did not study agents' ability to compose meaningful words to denote referents, i.e. they did not address the second level of the duality of patterning.

One of the goals of emergent communication research is to develop machines that can interact with humans. As a result, a variety of referential game approaches ensure that the emergent language is as close to natural language. This can be achieved by adding a supervised image captioning objective to encourage agents to use natural language in order to solve their communicative tasks (Havrylov & Titov, 2017; Lazaridou et al., 2017). Other methods use constraints such as memory restrictions (Kottur et al., 2017) to act as an effective information bottleneck to increase interpretability and compositionality. While we purposefully chose a graphical sensory-motor system to ease the visualization of the emerging language, we do not inject prior knowledge or pressures to facilitate the emergence of an iconic language. Our produced utterances are completely arbitrary. This fundamentally differentiates our work from Mihai & Hare (2021b) that trains agents to communicate via sketches replicating the visual referents they name. Note also that their drawing setup does not include dynamical motor primitives and utterances are directly optimized in image space allowing gradients to back-propagate from listener to speaker. Finally, they do not consider contrastive learning. To our knowledge, CURVES is the first contrastive deep-learning algorithm successfully applied to a referential game.

There is a large body of work exploring the factors that promote compositionally in emerging languages (Kottur et al., 2017; Li & Bowling, 2019; Rodríguez Luna et al., 2020; Ren et al., 2020; Chaabouni et al., 2020; Gupta et al., 2020). In this context, a crucial question is how to actually measure it in the first place (Mu & Goodman, 2021). To this end, (Choi et al., 2018) proposes to measure communicative performances on unseen compositions of known objects as a way to evaluate compositionality. However, it has been shown that a good performance in this test may be achieved without leveraging any actual compositionality in language (**?**Chaabouni et al., 2020). Thus, others instead compute topographic similarities (Brighton & Kirby, 2006), measuring the correlation between distances in the utterance space (distance between signs) and distances in the referents space (such as the cosine similarity between the embeddings of objects) (Lazaridou et al., 2018). In this paper we propose to do both and study 1) the generalization to unseen combinations of abstract features and 2) topographic measures based on the Hausdorff distances between utterances denoting composition and utterances denoting isolated features.

Contributions. This paper introduces:

- The Graphical Referential Game (GREG): a variation of the referential game to study the formation of signs from a graphical sensory-motor system.
- CURVES: an algorithmic solution to GREG, consisting of a contrastive multimodal encoder coupled with a generative model enabling the emergence of a graphical language.
- A systematic study of CURVES's generalization performances on compositions of features never seen during training in a simplified control setting and a more per-

ceptually challenging one.
- A complementary analysis of the compositionality of the emerging graphical language measuring the Hausdorff distance between utterances denoting compositions and utterances denoting their constituents.

5.2 Graphical Referential Games

Graphical referential game.

We consider a group of two agents playing a fixed number of referential games, each time alternating their roles (speaker or listener). We will consider two versions of the game: the *discriminative* and the *descriptive*. The *discriminative* game consists in presenting n objects R, called referents, to a speaker S and a listener L. At the beginning of each game, the target $r^\star \in R$ is assigned to the speaker. Given this target referent $r^\star$, S produces an utterance (u) to designate it. Based on the produced utterance u, L selects a referent ($\hat{r}$) in R. The game outcome o is a success if the selected referent ($\hat{r} \in R$) matches the target $r^\star$. The *descriptive* game is the same, except that the speaker is only provided with $r^\star$ and does not see the context R.

Additionally, we will consider scenarios, where agents perceive referents from different perspectives. In this case, the speaker perceives the referents R as $\tilde{R}_S$ and its target $r^\star$ as $r^\star_S$. Similarly, the listener perceives the referents R as $\tilde{R}_L$ and selects a referent $\hat{r}$ among it.

Sensory-motor drawing system.

Utterances are produced by a sensory-motor system $M : \mathbb{R}^m \to \mathcal{U} \subset \mathbb{R}^{D\times D}$ mimicking an arm drawing sketches displayed in Figure 5.2(a). The arm motion is derived from Dynamical Motor Primitives (DMPs) (Schaal, 2006). The DMP is parametrized by a command vector $c \in \mathbb{R}^{20}$. It converts c into a 2-dimensional drawing trajectory T made of 10 coordinates $T = \{v_i\}_{i=0,\dots,9}$. This trajectory is then fed to a Differentiable Sketching model (Mihai & Hare, 2021a) generating an $D \times D$ image (in our implementation, $D = 52$).

Referents.

Referents are compositions of orthogonal vector features (one-hot vectors). Given a set of m orthogonal features F_m, we define the set of all possible referents as $\mathcal{R}_m = \{\sum_{f\in S} f | S \subseteq F_m\}$. The subset of referents made of exactly k features are thus: $\mathcal{R}^k_m = \{\sum_{f\in S} f | S \subseteq F_m, |S| = k\}$. In our experiments, we fix $m = 5$.

From these orthogonal referents, we propose to generate objects made of digit images sampled from the MNIST dataset (LeCun et al., 1998). More precisely, we define the stochastic mapping $\Phi : \mathcal{R}_m \to \tilde{\mathcal{R}}_m$ that maps each feature $f \in F_m$ to a digit class in the MNIST dataset. For each feature in a referent, we sample a random instance from the corresponding class and randomly place it on a 4×4 grid such that no number overlap.

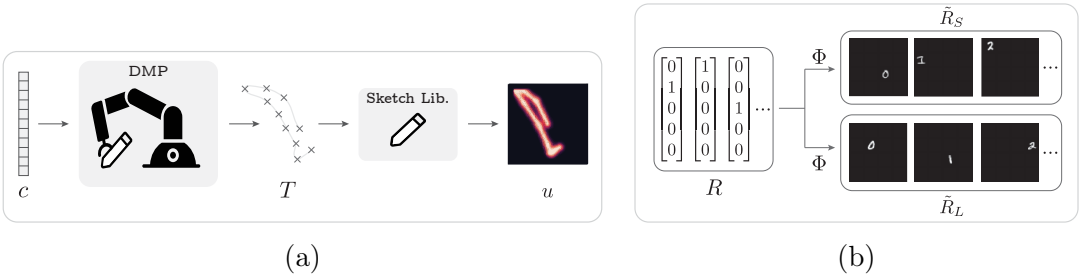


Figure 5.2: (a) **Sketching sensory-motor system**: The sensory-motor system imitates a robotic arm drawing a sketch on a 2D plan. DMPs first convert a continuous command c into a sequence of coordinates T. This trajectory is then rendered as a 52×52 graphical utterance thanks to a differentiable sketching library. (b) **Referent transformation:** An example of a one-hot context R being transformed into two contexts $\tilde{R}_S$ and $\tilde{R}_L$ by the stochastic transformation Φ. The two contexts are different perspectives of the same objects.

Note that the listener and speaker can perceive different realizations of Φ, in this case, we say that they see different *perspectives* of the referents.

We use this formalism to instantiate three settings of the Graphical Referential Game (GREG):

- *one-hot*: where referents are one-hot vectors $r \in \mathcal{R}_m$.
- *visual-shared*: where referents are MNIST digits $r \in \tilde{\mathcal{R}}_m$ and agents share the same perspective: $\tilde{R}_S = \tilde{R}_L$.
- *visual-unshared* where referents are MNIST digits $r \in \tilde{\mathcal{R}}_m$ and agents have different perspectives of referents in their contexts $\tilde{R}_S \neq \tilde{R}_L$.

Objectives.

This paper investigates how a group of two agents can agree on a shared compositional language to denote referents, given a continuous sensorimotor system to produce utterances. Beyond the game's success, we evaluate the emerging language along two dimensions.

Coherence. First, we measure the coherence of the emerging lexicon. As utterances are 2-dimensional paths, similarity can be calculated using the Hausdorff distance between sequences of coordinates. The Hausdorff distance d_H is the maximum distance from any coordinate in a trajectory to the closest coordinate in the other: $d_H(T_1, T_2) = \max\{\sup_{v \in T_1} d(v, T_2), \sup_{v' \in T_2} d(T_1, v')\}$. In particular, we compute the following metrics.

- **Agents Coherence (A-coherence):** For a given referent r with the same perspective for all agents, measure the mean pairwise similarity between each agent's utterance.
- **Perspective Coherence (P-coherence)**: For a given agent and a given referent r, measure the mean pairwise similarity between utterances produced from different perspectives (different instances of $\Phi(r)$).
- **Referents Coherence (R-coherence)**: For a given agent, measure the mean pairwise similarity between utterances produced for different referents.

Compositionality. The second dimension of our evaluation explores the compositional properties of the emerging language. To this end, we first evaluate the generalization performances of our group to compositional referents never seen during training. More specifically, we train agents on $\mathcal{R}_{\text{train}} = \mathcal{R}_5^1$ (referents made of one feature) and test them on $\mathcal{R}_{\text{test}} = \mathcal{R}_5^2$ (referents made of two features). For visuals about compositional referents, see Suppl. Section B.1.1. We use the success rate SR to monitor the performances. However, a satisfactory success rate on this testing set does not necessarily imply that the emerging graphical language is in fact compositional. Agents could, for instance, denote compositional referents using newly invented signs.

To complement this analysis, we thus decide to estimate to what extent utterances denoting compositional referents are actually made of utterances denoting their constituents. To this end, we introduce a topographic score ρ that quantifies how an utterances denoting a compositional referent made of feature i and j ($u(r_{ij})$) is actually closer to utterances denoting isolated features $u(r_i)$ or $u(r_j)$ than the utterance naming other compositional referents ($u(r_{xy})$, $x \neq i, y \neq j$). For a detailed derivation of metric ρ, see Suppl. Section B.1.2.

5.3 CURVES

CURVES is an energy-based approach that relies on two mechanisms:

1. The contrastive learning of an energy landscape $E(r, u)$ is defined as the cosine similarity between utterance and referent embeddings.
2. The generation of an utterance that maximizes the energy for a given target referent $r_S^\star$.

Agents modules and interactions. Each agent $A \in \{A_1, A_2\}$ trains a contrastive model to learn utterance and referent representations. Its dual encoder (f_A, g_A) maps utterances and referents in a shared d-dimensional latent space: $f_A(\cdot, \theta_{fA}) : \mathcal{R}_m \to \mathbb{R}^d$ and $g_A(\cdot, \theta_{gA}) : \mathcal{U} \to \mathbb{R}^d$ such that $z_{rA} = f_A(r)$ and $z_{uA} = g_A(u)$, as displayed in Figure 5.3(a). The energy landscape for each agent is therefore: $E_A(r, u) = \cos(f_A(r), g_A(u))$

A given referential game unfolds as follows. Agents are randomly attributed roles, for instance A_1 is the speaker $A_1 \leftarrow S$ and A_2 is the listener $A_2 \leftarrow L$. The speaker is given a context $\tilde{R}_S$ and a target referent perceived as $r_S^\star$ to produce an utterance $\hat{u}$ intending to maximize $E_S(r_S^\star, u)$. The listener observes $\hat{u}$ and selects referent $\hat{r}$ in context $\tilde{R}_L$ that maximizes $E_L = (r, \hat{u})$:

$$\begin{cases} \hat{u} \approx u^\star = \underset{u \in \mathcal{U}}{\text{argmax}}\ E_S(r_S^\star, u) \quad \text{(utterance generation from speaker)} \\ \hat{r} = \underset{r \in \tilde{R}_L}{\text{argmax}}\ E_L(r, \hat{u}) \quad \text{(referent selection from listener)} \end{cases} \tag{5.1}$$

The outcome of the game is then $o = \mathbb{1}_{[\hat{r}=r^\star]} - b$ where b is a baseline parameter representing the mean success across previous games.

Contrastive representation learning in referential games. For a given context R, agents are randomly assigned their roles and play $n = |R|$ games. During these n games, roles are fixed and the speaker agent successively selects each referent of the context $\tilde{R}_S$ as the target $r_S^\star$. During interactions, the speaker collects data $\{(r_S^i, u^i, o^i)\}_{i=1,\dots,n}$

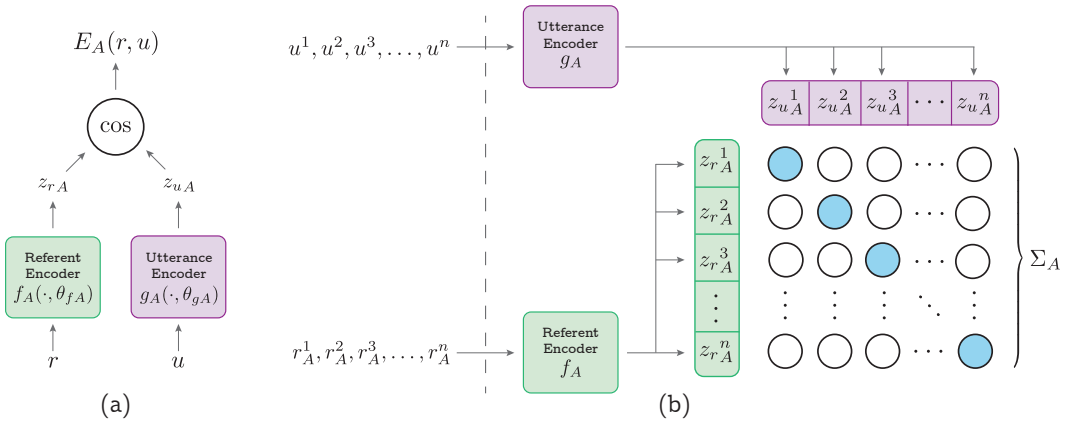


Figure 5.3: (a) **Agents's dual encoder architecture.** Referents and utterances are mapped to a share latent space. The energy between a referent r and an utterance u is computed as the cosine similarity between their respective embeddings. (b) **Cosine similarity matrix update from collected samples.** Agents compute the energy for all referents and utterances it collected to form the squared matrix Σ_A. During contrastive updates agents maximize blue circles and minimize white ones.

while the listeners observes $\{(u^i, r_L^i)\}_{i=1,\ldots,n}$. From the collected data each agent can compute the squared cosine similarity matrices Σ_A whose elements are $(\Sigma_A)_{i,j} = E_A(r_A^i, u^j)$ as shown in Figure 5.3(b). Contrastive updates are then performed using the objective J_A that applies *Cross Entropy* (CE) on the i-th row and i-th column of Σ_A.

$$J_A(\Sigma_A, i) = [CE((\Sigma_A)_{i,1:n}, e_i) + CE((\Sigma_A)_{1:n,i}, e_i)]/2, \tag{5.2}$$

e_i being a one-hot vector of size n with value 1 at index i. Depending on the role of the agent, J_A is instantiated either as J_S (speaker) or J_L (listener). Thus, the speaker updates its representation using the outcomes o_i of the games (reinforcing the successful associations while decreasing the unsuccessful ones):

$$\underset{\theta_{f_S},\theta_{g_S}}{\text{minimize}} \sum_{i=1}^{n} o_i J_S(\Sigma_S, i) \tag{5.3}$$

On the other hand, the listener needs to make sure that its selection matches the speaker's referent (Steels, 2015) and hence always increases associations (no matter the games' outcomes):

$$\underset{\theta_{f_L},\theta_{g_L}}{\text{minimize}} \sum_{i=1}^{n} J_L(\Sigma_L, i) \tag{5.4}$$

Speaker's utterance optimization. For the speaker agent, producing an utterance is formalized as maximizing the cosine similarity between the embeddings of a given referent and an utterance produced by our sensory system $u = M(c)$ from motor command c. Since M is fully differentiable, we inject the sensory-motor constraint in equation 5.1 and seek for the optimal motor command $c^\star$ using gradient ascent:

$$c^\star = \underset{c\in\mathbb{R}^p}{\text{argmax}}\ E(r_S^\star, M(c)) \tag{5.5}$$

Since the optimization is only performed from the energy between the produced utterance and the target referent $r_S^\star$ we call it *descriptive*. In practice, we do not have any guarantee to reach $c^\star$ and only approach it.

Alternatively, we propose to vary the generation conditions to a *discriminative* scenario where the speaker also perceives the context $\tilde{R}_S$ during production. This is achieved by finding the motor command that minimizes the cross entropy given a target referent $r_S^\star$ and its context $\tilde{R}_S$:

$$c^\star = \underset{c \in \mathbb{R}^p}{\operatorname{argmin}}\ CE(\sigma_S, e_{r_S^\star}) \tag{5.6}$$

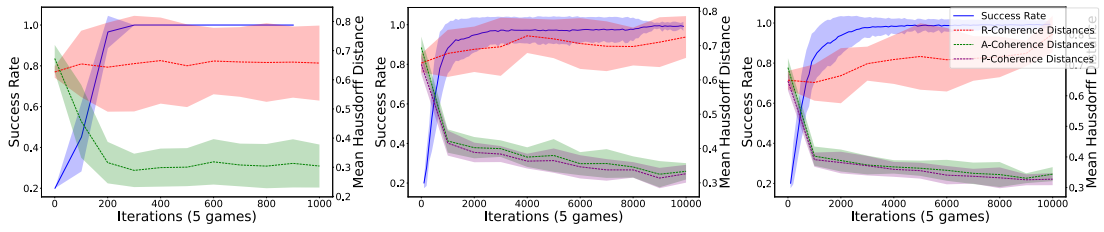
where σ_S is the vector with coordinates $\sigma_{Si} = [E(r^i, M(c))]_{r^i \in \tilde{R}_S}$ and $e_{r_S^\star}$ is the one-hot vector of size $|\tilde{R}_S|$ with value 1 at the position of $r_S^\star$ in $\tilde{R}_S$. This discriminative generation process is only used at test time when investigating CURVES's generalization capabilities.

5.4 Experiments and Results

This section focuses first on CURVES's training dynamics as agents interact in GREG before showcasing its ability to generalize to compositional referents that were never seen during training. We finally evaluate the compositional structure of the emerging graphical language by providing visuals of utterances and computing topographic scores defined in Section 5.2. Each of these studies is carried out with one-hot, shared-visual, and unshared-visual referents as explained in Section 5.2. Training and testing metrics correspond to the mean and standard deviation computed from training pairs of agents on 10 seeds.

Do agents converge to a shared graphical language?

Figure 5.4 displays the training performances of a group of two agents interacting in GREG. For each referent type, the group reaches a perfect success rate of SR = 1. Moreover, a group starts to converge when inter-agent and inter-perspective coherence distances decrease. This correlation is proof of emergent communication as it indicates that agents start agreeing on signs to denote referents. Finally, the constant (for one-hot referent) and increasing (for visual referents) values of the R-coherence suggest that agents use distinct signs to name referents.



(a) (b) (c)

Figure 5.4: **Training success rate and Coherence distances** (a) one-hot referents (b) visual-shared referents (c) visual-unshared referents.

An example of an emerging lexicon describing visual referents produced by agents trained on unshared perspectives can be visualized in Figure 5.5. Other visualizations

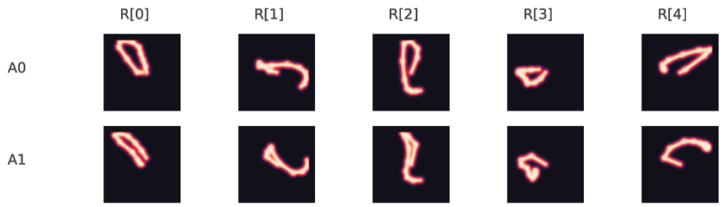


Figure 5.5: **Instance of an emerging lexicon.** Utterances are produced by a group of agents trained with unshared perspectives (1 seed). The perspective for each referent is chosen randomly.

for one-hot and shared visual referents are available in Suppl. Section B.2.2. We also provide illustrations of P-coherence in Suppl. Section B.2.3.

Are agents able to generalize to compositional referents?

Table 5.1 exposes the generalization performances of group of agents evaluated on referents $r \in \mathcal{R}_5^2$. During an evaluation, the context is exhaustive and contains all the combinations of 2 features: $|R| = 10$. We compare the success rates to a *random* baseline where the listener always selects the referent $\hat{r}_L$ randomly no matter the utterance ($\text{SR}_{\text{random}} = 0.1$). We also introduce a *1-feature* baseline where the speaker produces an utterance u that only denotes one of the two features contained in $r_S^\star$ and the listener randomly selects one of the four combinations containing the communicated feature ($\text{SR}_{\text{1-feat}} = 0.25$). The success rates for all referent types are significantly higher than the baseline values suggesting that agents are indeed able to communicate about compositional referents. Generalization performances are nearly perfect with one-hot referents but they decrease in visual settings. This performance gap can be explained by the extra difficulty of adding inter-perspective variability to the multi-agent interaction dynamic during the contrastive learning of referent representations. The better success rates obtained in auto-learning (where a single agent plays both the speaker and the listener roles) provided in Suppl. Section B.2.1 seem to corroborate this hypothesis. Surprisingly, we observe that success rates for descriptive (Eq. 5.5) and discriminative (Eq.5.6) generation are very similar. This suggests that optimizing utterances so as to minimize their energy between non-targeted compositional referents ($r \in R, r \neq r^\star$) does not improve generalization performances.

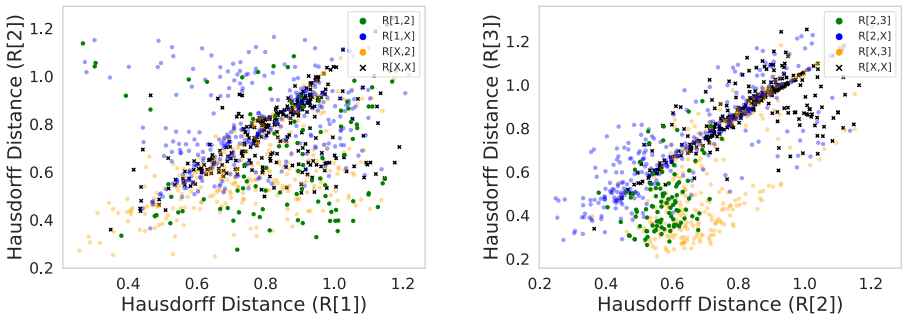
Referents	**Descriptive** SR	**Discriminative** SR
One-hot	0.99 ± 0.01	0.99 ± 0.01
Visual-shared	0.57 ± 0.04	0.56 ± 0.03
Visual-unshared	0.39 ± 0.02	0.40 ± 0.02

Table 5.1: **Generalization performances.** Success rates evaluated on exhaustive context $|R| = 10$ with referents $r \in \mathcal{R}_5^2$ for both generative (Eq. 5.5) and discriminative (Eq.5.6) utterance generation.

Is the emerging language compositional?

To investigate the compositionality of utterances we propose the topographic maps

displayed in Figure 5.6. Each point in a topographic map is an utterance naming a compositional referent $r \in \mathcal{R}_5^2$ and has coordinate $(d_H(u(r_i), \cdot), d_H(u(r_j), \cdot))$. If utterances naming the composition of two features are indeed the compositions of the utterances used to denote each of the isolated features, we expect them to land in the bottom left of the topographic maps. Figure 5.6 shows that some utterances for compositional referents are indeed close in Haussdorf distance to the utterances denoting the isolated constituent features (Figure 5.6(b)) but others are not (Figure 5.6(a)).



(a) (b)

Figure 5.6: **Topographic map examples for a single seed in unshared-visual referents setting** (a) Corresponding to the worst topographic score $\rho = -0.113$ (b) Corresponding to the best topographic score $\rho = 0.203$. Each utterance names a compositional referent and is colored in blue if it contains feature i, orange if it contains feature j, green if it contains both, and black if it contains none.

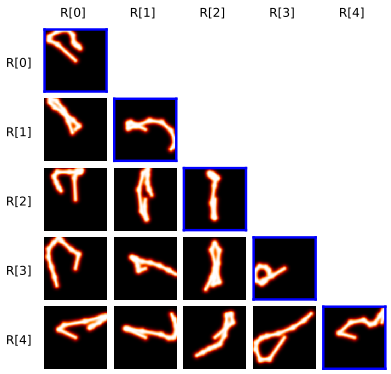


Figure 5.7: **Matrix of compositions.** Blue frames represent utterances generated for a perspective in $\mathcal{R}_5^1$, other utterance denote the corresponding compositions in $\mathcal{R}_5^2$

Unfortunately, the feature maps do not allow us to draw strong conclusions about the composition properties of the emerging language. It is hard to tell if agents are indeed composing utterances or if the Haussdorf distance simply does not capture compositionality. This seems to be verified by additional topographic maps provided in Suppl. Section B.2.4. In particular, the topographic maps for one-hot referents (Figure B.10) indicate that strong generalization performances can be achieved by producing utterances

that are not necessarily close to the isolated feature utterances. This difficulty in evaluating compositionality can be experienced visually thanks to Figure 5.7 which displays a matrix of composition for unshared-visual referents. More instances of compositions matrices are available in Suppl. Section B.2.5.

Are representations compositional?

Finally, if compositionality is visually hard to analyze in graphical space, it seems to be much more apparent in the utterance and referent embedding spaces. Figure 5.8 shows that the embeddings for compositional referents as well as the embedding of the utterances naming them are indeed close to the embeddings of their constituents. This is not surprising since this is the space in which our energy landscape is learned.

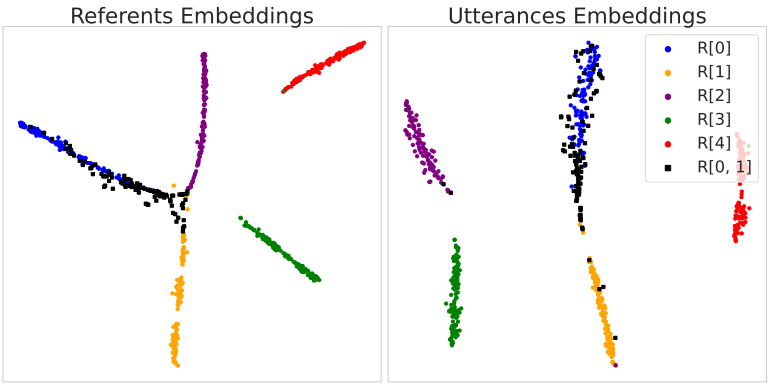


Figure 5.8: **T-sne of utterance and referent embeddings.** Embeddings are computed for 100 perspectives of referents. Training conditions are unshared visual referents. Additional t-snes are provided in Suppl. Section B.2.6

5.5 Discussion and future work

This work formalizes GREG: a new referential game where two agents must communicate via a continuous sensory-motor system imitating a robotic arm drawing sketches. To tackle GREG, we propose CURVES: a contrastive representation learning algorithm inspired by early language game contrastive implementation that scales to high dimensional signals. CURVES allows a group of two agents two converge on a shared graphical language in contexts where referents are one-hot vectors or images of MNIST digits. Additionally, the representations that agents learn enable them to communicate about compositional referents never encountered during training. Despite the visualizable nature of graphical signs, compositions of utterances are hard to identify. Our proposed analysis based on the Hausdorff distance could not allow us to draw systematic conclusions. On the other hand, compositions are salient in the space of embeddings.

Future work may look into finding other metrics or evaluation strategies to investigate the composition of utterances in more depth. An analysis of the impact of the sensory-motor constraints on the topology of graphical signs could also provide valuable insight

into the factors facilitating the emergence of a compositional graphical language. CURVES is agnostic to the modality used to represent utterances. As such, it could tackle other sensory-motor systems. The central element of CURVES lies in the contrastive learning of utterance-referent associations. In our implementation, we optmize utterances by maximizing this energy via gradient ascent. Much like CLIP opened many avenues for multi-modal generation, we could plug in more complex generative strategies such as diffusion models (Rombach et al., 2021; Saharia et al., 2022). Finally, more realistic visual referents and the impact of training larger groups of agents on generalization could be investigated in GREG.

Part II

Exploitation of Cultural Models

Chapter 6

Foundations: Cultural Internalisation - Towards Vygotskian Autotelic Agents

Contents

6.1 Developmental Machine Learning and Intrinsic motivations

Building autonomous machines that can explore large environments, discover interesting interactions and learn open-ended repertoires of skills is a long-standing goal in artificial intelligence. Humans are remarkable examples of this lifelong, open-ended learning. They learn to recognize objects and crawl as infants, then learn to ask questions and interact with peers as children. Across their lives, humans build a large repertoire of diverse skills from a virtually infinite set of possibilities. What is most striking, perhaps, is their ability to invent and pursue their own problems, using internal feedback to assess completion. We would like to build artificial agents able to demonstrate equivalent lifelong learning abilities.

We can think of two approaches to this problem: developmental approaches, in particular developmental robotics, and reinforcement learning (RL). Developmental robotics takes inspirations from artificial intelligence, developmental psychology and neuroscience to model cognitive processes in natural and artificial systems Asada et al. (2009); Cangelosi & Schlesinger (2015). Following the idea that intelligence should be *embodied*, robots are often used to test learning models. Reinforcement learning, on the other hand, is the field interested in problems where agents learn to behave by experiencing the consequences of their actions under the form of rewards and costs. As a result, these agents are not explicitly taught, they need to learn to maximize cumulative rewards over time by trial-and-error Sutton & Barto (2018). While developmental robotics is a field oriented towards answering particular questions around sensorimotor, cognitive and social development (e.g. how can we model language acquisition?), reinforcement learning is a field organized around a particular technical framework and set of methods.

Now powered by deep learning optimization methods leveraging the computational efficiency of large computational clusters, RL algorithms have recently achieved remarkable results including, but not limited to, learning to solve video games at a super-human level Mnih et al. (2015), to beat chess and go world players Silver et al. (2016), or even to control stratospheric balloons in the real world Bellemare et al. (2020).

Although standard RL problems often involve a single agent learning to solve a unique task, RL researchers extended RL problems to *multi-goal RL problems*. Instead of pursuing a single goal, agents can now be trained to pursue goal distributions Kaelbling (1993); Sutton et al. (2011); Schaul et al. (2015a). As the field progresses, new goal representations emerge: from the specific goal states to the high-dimensional goal images or the abstract language-based goals Luketina et al. (2019). However, most approaches still fall short of modeling the learning abilities of natural agents because they train them to solve predefined sets of tasks, via external and hand-defined learning signals.

Developmental robotics directly aims to model children learning and, thus, takes inspiration from the mechanisms underlying autonomous behaviors in humans. Most of the time, humans are not motivated by external rewards but spontaneously explore their environment to discover and learn about what is around them. This behavior seems to be driven by *intrinsic motivations* (IMs) a set of brain processes that motivate humans to explore for the mere purpose of experiencing novelty, surprise or learning progress Berlyne (1966a); Gopnik et al. (1999); Kidd & Hayden (2015a); Oudeyer & Smith (2016); Gottlieb & Oudeyer (2018a).

The integration of IMs into artificial agents thus seems to be a key step towards autonomous learning agents Schmidhuber (1991c); Kaplan & Oudeyer (2007). In developmental robotics, this approach enabled sample efficient learning of high-dimensional motor skills in complex robotic systems Santucci et al. (2020), including locomotion Baranes & Oudeyer (2013); Martius et al. (2013), soft object manipulation Rolf & Steil (2013); Nguyen & Oudeyer (2014), visual skills Lonini et al. (2013) and nested tool use in real-world robots Forestier et al. (2022). Most of these approaches rely on *population-based* optimization algorithms, non-parametric models trained on datasets of (policy, outcome) pairs. Population-based algorithms cannot leverage automatic differentiation on large computational clusters, often demonstrate limited generalization capabilities and cannot easily handle high-dimension perceptual spaces (e.g. images) without hand-

defined input pre-processing. For these reasons, developmental robotics could benefit from new advances in deep RL.

Recently, we have been observing a convergence of these two fields, forming a new domain that we propose to call *developmental reinforcement learning*, or more broadly *developmental artificial intelligence*. Indeed, RL researchers now incorporate fundamental ideas from the developmental robotics literature in their own algorithms, and reversely developmental robotics learning architecture are beginning to benefit from the generalization capabilities of deep RL techniques. These convergences can mostly be categorized in two ways depending on the type of intrinsic motivation (IMs) being used Oudeyer & Kaplan (2007):

- **Knowledge-based IMs** are about prediction. They compare the situations experienced by the agent to its current knowledge and expectations, and reward it for experiencing dissonance (or resonance). This family includes IMs rewarding prediction errors Schmidhuber (1991c); Pathak et al. (2017), novelty Bellemare et al. (2016); Burda et al. (2019); Raileanu & Rocktäschel (2020), surprise Achiam & Sastry (2017), negative surprise Berseth et al. (2019), learning progress Lopes et al. (2012); Kim et al. (2020) or information gains Houthooft et al. (2016), see a review in Linke et al. (2020). This type of IM is often used as an auxiliary reward to organize the exploration of agents in environments characterized by sparse rewards. It can also be used to facilitate the construction of world models Lopes et al. (2012); Kim et al. (2020); Sekar et al. (2020).

- **Competence-based IMs**, on the other hand, are about control. They reward agents to solve self-generated problems, to achieve self-generated goals. In this category, agents need to represent, select and master self-generated goals. As a result, competence-based IMs were often used to organize the acquisition of repertoires of skills in task-agnostic environments Baranes & Oudeyer (2010, 2013); Santucci et al. (2016); Forestier & Oudeyer (2016); Nair et al. (2018b); Warde-Farley et al. (2019); Colas et al. (2019a); Blaes et al. (2019); Pong et al. (2020); Colas et al. (2020b). Just like knowledge-based IMs, competence-based IMs organize the exploration of the world and, thus, might be used to train world models Baranes & Oudeyer (2013); Chitnis et al. (2021) or facilitate learning in sparse reward settings Colas et al. (2018). We propose to use the adjective **autotelic**, from the Greek *auto* (self) and *telos* (end, goal), to characterize agents that are intrinsically motivated to represent, generate, pursue and master their own goals (i.e. that are both intrinsically motivated and goal-conditioned).

RL algorithms using *knowledge-based* IMs leverage ideas from developmental robotics to solve standard RL problems. On the other hand, RL algorithms using competence-based IMs organize exploration around self-generated goals and can be seen as targeting a developmental robotics problem: the *open-ended and self-supervised acquisition of repertoires of diverse skills*.

Intrinsically Motivated Goal Exploration Processes (IMGEP) is the family of autotelic algorithms that bake competence-based IMs into learning agents Forestier et al. (2022). IMGEP agents generate and pursue their own goals as a way to explore their environment,

discover possible interactions and build repertoires of skills. This framework emerged from the field of developmental robotics Oudeyer & Kaplan (2007); Baranes & Oudeyer (2009a, 2010); Rolf et al. (2010) and originally leveraged population-based learning algorithms (POP-IMGEP) Baranes & Oudeyer (2009b, 2013); Forestier & Oudeyer (2016); Forestier et al. (2022).

Recently, goal-conditioned RL agents were also endowed with the ability to generate and pursue their own goals and learn to achieve them via self-generated rewards. We call this new set of autotelic methods RL-IMGEPs. In contrast, one can refer to externally-motivated goal-conditioned RL agents as RL-EMGEPs.

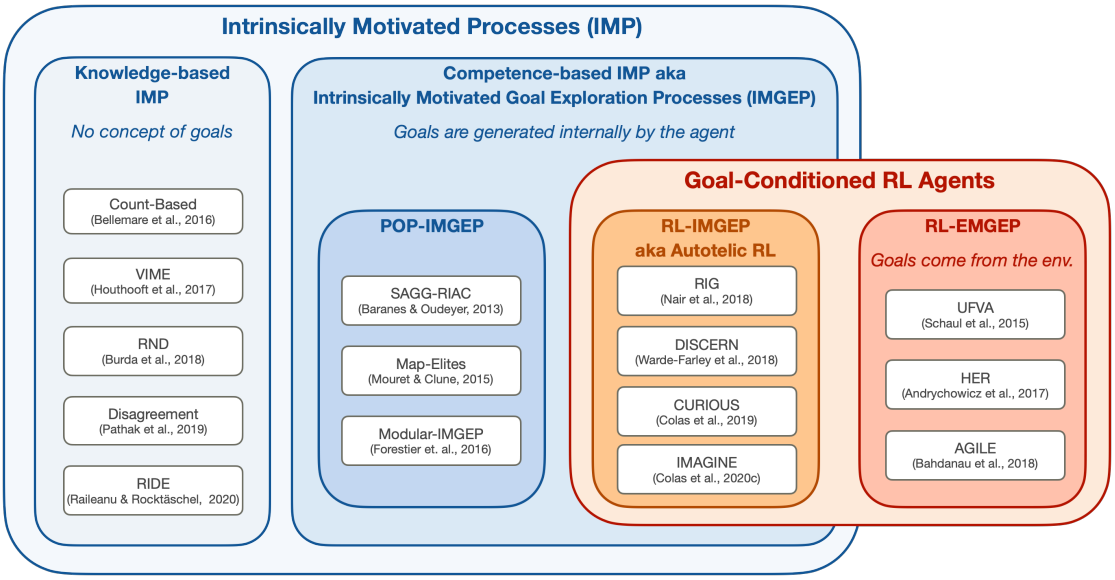


Figure 6.1: A typology of intrinsically-motivated and/or goal-conditioned RL approaches. POP-IMGEP, RL-IMGEP and RL-EMGEP refer to *population-based intrinsically motivated goal exploration processes*, *RL-based IMGEP* and *RL-based externally motivated goal exploration processes* respectively. POP-IMGEP, RL-IMGEP and RL-EMGEP all represent goals, but knowledge-based IMs do not. While IMGEPs (POP-IMGEP and RL-IMGEP) generate their own goals, RL-EMGEPs require externally-defined goals. This paper is interested in RL-IMGEPs, autotelic methods at the intersection of *goal-conditioned RL agents* and *intrinsically motivated processes* that train learning agents to generate and pursue their own goals with goal-conditioned RL algorithms.

6.2 From Reinforcement to Autotelic Learning

Todo:*select some concepts to include in the introduction* **Todo:***clean section and sub*

This paper proposes a formalization and a review of the RL-IMGEP algorithms at the convergence of RL methods and developmental robotics objectives. Figure 6.1 proposes a visual representation of intrinsic motivations approaches (knowledge-based IMs vs competence-based IMs or IMGEPs) and goal-conditioned RL (externally vs intrinsically motivated). Their intersection is the family of autotelic algorithms that train agents to generate and pursue their own goals by training goal-conditioned policies.

We define goals as the combination of a compact goal representation and a goal-achievement function to measure progress. This definition highlights new challenges for autonomous learning agents. While traditional RL agents only need to learn to achieve goals, RL-IMGEP agents also need to learn to represent them, to generate them and to measure their own progress. After learning, the resulting goal-conditioned policy and its associated goal space form a *repertoire of skills*, a repertoire of behaviors that the agent can represent and control. We believe organizing past goal-conditioned RL algorithms at the convergence of developmental robotics and RL into a common classification and towards the resolution of a common problem will help organize future research.

Definitions

- ***Goal***: "a cognitive representation of a future object that the organism is committed to approach Elliot & Fryer (2008)." In RL, this takes the form of a (embedding, goal-achievement function) pair, see Section 2.2.1.
- ***Skill***: the association of a goal and a policy to reach it, see Section 6.2.1.
- ***Goal-achievement function***: a function that measures progress towards a goal (also called goal-conditioned reward function), see Section 2.2.1.
- ***Goal-conditioned policy***: a function that generates the next action given the current state and the goal, see Section 6.2.1.
- ***Autotelic***: from the Greek *auto* (self) and *telos* (end, goal), characterizes agents that generate their own goals and learning signals. In is equivalent to *intrinsically motivated and goal-conditioned*.

6.2.1 The Intrinsically Motivated Skills Acquisition Problem and the RL-IMGEP Framework

This section builds on the multi-goal RL problem to formalize the *intrinsically motivated skills acquisition problem*, in which goals are not externally provided to the agents but must be represented and generated by them (Section 6.2.1). The following section discusses how to evaluate competency in such an open problem (Section 6.2.1). Finally, we then propose an extension of the goal-conditioned RL framework to tackle this problem: *rl-based intrinsically motivated goal exploration process* framework (RL-IMGEP, Section 6.2.2).

The Intrinsically Motivated Skills Acquisition Problem

In the *intrinsically motivated skills acquisition problem*, the agent is set in an open-ended environment without any pre-defined goal and needs to acquire a repertoire of skills. Here, a skill is defined as the association of a goal embedding z_g and the policy to reach it Π_g. A repertoire of skills is thus defined as the association of a repertoire of goals $\mathcal{G}$ with a goal-conditioned policy trained to reach them $\Pi_{\mathcal{G}}$. The intrinsically motivated skills acquisition problem can now be modeled by a reward-free MDP $\mathcal{M} = \{\mathcal{S}, \mathcal{A}, \mathcal{T}, \rho_0\}$ that only characterizes the agent, its environment and their possible interactions. Just

like children, agents must be autotelic, i.e. they should learn to represent, generate, pursue and master their own goals.

Evaluating RL-IMGEP Agents.

Evaluating agents is often trivial in reinforcement learning. Agents are trained to maximize one or several pre-coded reward functions — the set of possible interactions is known in advance. One can measure generalization abilities by computing the agent's success rate on a held-out set of testing goals. One can measure exploration abilities via several metrics such as the count of task-specific state visitations.

In contrast, autotelic agents evolve in open-ended environments and learn to represent and form their own set of skills. In this context, the space of possible behaviors might quickly become intractable for the experimenter, which is perhaps the most interesting feature of such agents. For these reasons, designing evaluation protocols is not trivial.

The evaluation of such systems raises similar difficulties as the evaluation of task-agnostic content generation systems like Generative Adversarial Networks (GAN) Goodfellow et al. (2014) or self-supervised language models Devlin et al. (2019); Brown et al. (2020). In both cases, learning is *task-agnostic* and it is often hard to compare models in terms of their outputs (e.g. comparing the quality of GAN output images, or comparing output repertoires of skills in autotelic agents).

One can also draw parallel with the debate on the evaluation of open-ended systems in the field of *open-ended evolution* Hintze (2019); Stanley & Soros (2016); Stanley (2019). In both cases, a *good* system is expected to generate more and more original solutions such that its output cannot be predicted in advance. But what does *original* mean, precisely? Stanley & Soros (2016) argues that subjectivity has a role to play in the evaluation of open-ended systems. Indeed, the notion of *interestingness* is tightly coupled with that of *open-endedness*. What we expect from our open-ended systems, and of our RL-IMGEP agents in particular, is to generate more and more behaviors that *we* deem interesting. This is probably why the evaluation of content generators often include human studies. Our end objective is to generate interesting artefacts for us; we thus need to evaluate open-ended processes ourselves, subjectively.

Our end goal would be to interact with trained RL-IMGEP directly, to set themselves goals and test their abilities. The evaluation would need to adapt to the agent's capabilities. As Einstein said *"If you judge a fish by its ability to climb a tree, it will live its whole life believing that it is stupid."*. RL-IMGEP need to be evaluated by humans looking for their area of expertise, assessing the width and depth of their capacities in the world they were trained in. This said, science also requires more objective evaluation metrics to facilitate the comparison of existing methods and enable progress. Let us list some evaluation methods measuring the competency of agents via proxies:

- **Measuring exploration:** one can compute task-agnostic exploration proxies such as the entropy of the visited state distribution, or measures of state coverage (e.g. coverage of the high-level x-y state space in mazes) Florensa et al. (2018). Exploration can also be measured as the number of interactions from a set of *inter-*

esting interactions defined subjectively by the experimenter **?**e.g. interactions with objects in>imagine.

- **Measuring generalization:** The experimenter can subjectively define a set of relevant target goals and prevent the agent from training on them. Evaluating agents on this held-out set at test time provides a measure of generalization Ruis et al. (2020), although it is biased towards what the experimenter assesses as *relevant* goals.

- **Measuring transfer learning:** The intrinsically motivated exploration of the environment can be seen as a pre-training phase to bootstrap learning in a subsequent downstream task. In the downstream task, the agent is trained to achieve externally-defined goals. We report its performance and learning speed on these goals. This is akin to the evaluation of self-supervised language models, where the reported metrics evaluate performance in various downstream tasks Brown et al. (2020). In this evaluation setup, autotelic agents can be compared to task-specific agents. Ideally, autotelic agents should benefit from their open-ended learning process to outperform task-specific agents on their own tasks. This said, performance on downstream tasks remains an evaluation proxy and should not be seen as the explicit *objective* of the skill discovery phase. Indeed, in humans, skill discovery processes do not target any specific future task, but emerged from a natural evolutionary process maximizing reproductive success, see a discussion in Singh et al. (2010).

- **Opening the black-box:** Investigating internal representations learned during intrinsically motivated exploration is often informative. One can investigate properties of the goal generation system (e.g. does it generate out-of-distribution goals?), investigate properties of the goal embeddings (e.g. are they disentangled?). One can also look at the learning trajectories of the agents across learning, especially when they implement their own curriculum learning Florensa et al. (2018); Colas et al. (2019a); Blaes et al. (2019); Pong et al. (2020); Akakzia et al. (2021a).

- **Measuring robustness:** Autonomous learning agents evolving in open-ended environment should be robust to a variety of properties than can be found in the real-world. This includes very large environments, where possible interactions might vary in terms of difficulty (trivial interactions, impossible interactions, interactions whose result is stochastic thus prevent any learning progress). Environments can also include distractors (e.g. non-controllable objects) and various forms of non-stationarity. Evaluating learning algorithms in various environments presenting each of these properties allows to assess their ability to solve the corresponding challenges.

6.2.2 RL-Based Intrinsically Motivated Goal Exploration Processes

Until recently, the IMGEP family was powered by population-based algorithms (POP-IMGEP). The emergence of goal-conditioned RL approaches that generate their own goals gave birth to a new type of IMGEPs: the RL-based IMGEPs (RL-IMGEP). This section

builds on traditional RL and goal-conditioned RL algorithms to give a general definition of intrinsically motivated goal-conditioned RL algorithms (RL-IMGEP).

RL-IMGEP are intrinsically motivated versions of goal-conditioned RL algorithms. They need to be equipped with mechanisms to represent and generate their own goals in order to solve the intrinsically motivated skills acquisition problem, see Figure 6.2. Concretely, this means that, in addition to the goal-conditioned policy, they need to learn: 1) to represent goals g by compact embeddings z_g; 2) to represent the support of the goal distribution, also called *goal space* $\mathcal{Z}_\mathcal{G} = \{z_g\}_{g \in \mathcal{G}}$; 3) a goal distribution from which targeted goals are sampled $\mathcal{D}(z_g)$; 4) a goal-conditioned reward function $\mathcal{R}_\mathcal{G}$. In practice, only a few architectures tackle the four learning problems above.

In this survey, **we call *autotelic* any architecture where the agent selects its own goals (learning problem 3)**. Simple autotelic agents assume pre-defined goal representations (1), the support of the goals distribution (2) and goal-conditioned reward functions (4). As autotelic architectures tackle more of the 4 learning problems, they become more and more advanced. As we will see in the following sections, many existing works in goal-conditioned RL can be formalized as autotelic agents by including goal sampling mechanisms *within the definition of the agent*.

With a developmental perspective, one can reinterpret existing work through the autotelic RL framework. Let us take an example. The AGENT$_{57}$ algorithm automatically selects a parameter to balance the intrinsic and extrinsic rewards of the agent at the beginning of each training episode Badia et al. (2020a). The authors do not mention the concept of *goal* but instead present this mechanism as a form of reward shaping technique independent from the agent. With a developmental perspective, one can interpret the mixing parameter as a goal embedding. Replacing the sampling mechanism within the boundaries of the agent, AGENT$_{57}$ becomes autotelic. It is intrinsically motivated to sample and target its own goals; i.e. to define its own reward functions (here mixtures of intrinsic and extrinsic reward functions).

Algorithm 1 details the pseudo-code of RL-IMGEP algorithms. Starting from randomly initialized modules and memory, RL-IMGEP agents enter a standard RL interaction loop. They first observe the context (initial state), then sample a goal from their goal sampling policy. Then starts the proper interaction. Conditioned on their current goal embedding, they act in the world so as to reach their goal, i.e. to maximize the cumulative rewards generated by the goal-conditioned reward function. After the interaction, the agent can update all its internal models. It learns to represent goals by updating its goal embedding function and goal-conditioned reward function, and improves its behavior towards them by updating its goal-conditioned policy.

This surveys focuses on the mechanisms specific to RL-IMGEP agents, i.e. mechanisms that handle the representation, generation and selection of goals. These mechanisms are mostly orthogonal to the question of how to reach the goals themselves, which often relies on existing goal-conditioned algorithms, but can also be powered by imitation learning, evolutionary algorithms or other control and planning methods. Section 6.2.3 first presents a typology of goal representations used in the literature, before Sections 6.2.4 and 6.2.5 cover existing methods to learn to represent and prioritize goals respectively.

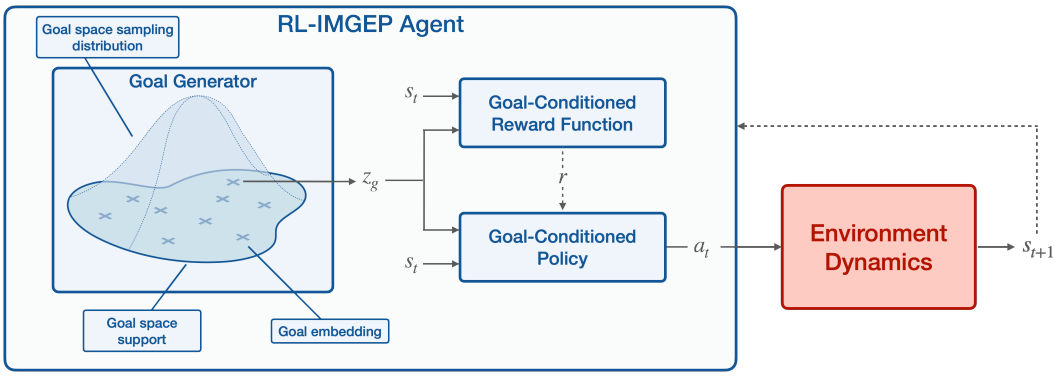
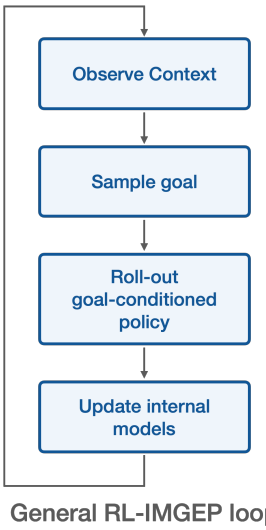


Figure 6.2: Representation of the different learning modules in a RL-IMGEP algorithm. In contrast, externally motivated goal exploration processes (RL-EMGEPs) only train the goal-conditioned policy and assume *external* goal generator and goal-conditioned reward function. Learning goal embeddings, goal space support and goal-conditioned reward functions are all about learning to *represent goals*. Learning a sampling distribution is about learning to *prioritize their selection*.

Algorithm 1: Autotelic Agent with RL-IMGEP

Require: environment $\mathcal{E}$

Initialize empty memory $\mathcal{M}$,goal-conditioned policy $\Pi_\mathcal{G}$, goal-conditioned reward $R_\mathcal{G}$,goal space $\mathcal{Z}_\mathcal{G}$, goal sampling policy GS.
loop
  Get initial state: $s_0 \leftarrow \mathcal{E}$.reset()
  Sample goal embedding $z_g = GS(s_0, \mathcal{Z}_\mathcal{G})$.
  Execute a roll-out with $\Pi_g = \Pi_\mathcal{G}(\cdot \mid z_g)$
  Store collected transitions $\tau = (s, a, s')$ in $\mathcal{M}$.
  Sample a batch of B transitions: $\mathcal{M} \sim \{(s, a, s')\}_B$.
  Perform Hindsight Relabelling $\{(s, a, s', z_g)\}_B$.
  Compute internal rewards $r = R_\mathcal{G}(s, a, s' \mid z_g)$.
  Update policy $\Pi_\mathcal{G}$ via RL on $\{(s, a, s', z_g, r)\}_B$.
  Update goal representations $\mathcal{Z}_\mathcal{G}$.
  Update goal-conditioned reward function $R_\mathcal{G}$.
  Update goal sampling policy GS.
end loop
return $\Pi_\mathcal{G}, R_\mathcal{G}, \mathcal{Z}_\mathcal{G}$



6.2.3 A Typology of Goal Representations in the Literature

Now that we defined the problem of interest and the overall framework to tackle it, we can start reviewing relevant approaches from the literature and how they fit in this framework. This section presents a typology of the different kinds of goal representations found in the literature. Each goal is represented by a pair: 1) a *goal embedding* and 2) a goal-conditioned reward function. Figure 6.3 also provides visuals of the main environments used by the autotelic approaches presented in this paper.

Goals as Choices Between Multiple Objectives

Goals can be expressed as a list of different objectives the agent can choose from.

Goal embedding. In that case, goal embeddings z_g are one-hot encodings of the current objective being pursued among the N objectives available. z_g^i is the i^{th} one-hot vector: $z_g^i = (\mathbb{1}_{j=i})_{j=[1..N]}$. This is the case in Oh et al. (2017); Mankowitz et al. (2018); Codevilla et al. (2018).

Reward function. The goal-conditioned reward function is a collection of N distinct reward functions $R_\mathcal{G}(\cdot) = R_i(\cdot)$ if $z_g = z_g^i$. In Mankowitz et al. (2018) and Chan et al. (2019), each reward function gives a positive reward when the agent reaches the corresponding object: reaching guitars and keys in the first case, monsters and torches in the second.

Goals as Target Features of States

Goals can be expressed as target features of the state the agent desires to achieve.

Goal embedding. In this scenario, a state representation function φ maps the state space to an embedding space $\mathcal{Z} = \varphi(\mathcal{S})$. Goal embeddings z_g are target points in $\mathcal{Z}$ that the agent should reach. In manipulation tasks, z_g can be target block coordinates Andrychowicz et al. (2017a); Nair et al. (2018a); Plappert et al. (2018); Colas et al. (2019a); Fournier et al. (2021); Blaes et al. (2019); Lanier et al. (2019); Ding et al. (2019); Li et al. (2020). In navigation tasks, z_g can be target agent positions **?**e.g. in mazes, >schaul2015universal,goalgan. Agent can also target image-based goals. In that case, the state representation function φ is usually implemented by a generative model trained on experienced image-based states and goal embeddings can be sampled from the generative model or encoded from real images Zhu et al. (2017); Codevilla et al. (2018); Nair et al. (2018b); Pong et al. (2020); Warde-Farley et al. (2019); Florensa et al. (2019); Venkattaramanujam et al. (2019); Lynch et al. (2020); Lynch & Sermanet (2020); Nair et al. (2020); Kovač et al. (2020).

Reward function. For this type of goals, the reward function $R_\mathcal{G}$ is based on a distance metric D. One can define a dense reward as inversely proportional to the distance between features of the current state and the target goal embedding: $R_g = R_\mathcal{G}(s|z_g) = -\alpha \times D(\varphi(s),\ z_g)$ **?**e.g. >nair2018visual. The reward can also be sparse: positive whenever that distance falls below a pre-defined threshold: $R_\mathcal{G}(s|z_g) = 1$ if $D(\varphi(s),\ z_g) < \epsilon$, 0 otherwise.

Goals as Abstract Binary Problems

Some goals cannot be expressed as target state features but can be represented by *binary problems*, where each goal expresses as set of constraint on the state (or trajectory) such that these constraints are either verified or not (binary goal achievement).

Goal embeddings. In binary problems, goal embeddings can be any expression of the set of constraints that the state should respect. Akakzia et al. (2021a); Ecoffet et al. (2021) both propose a pre-defined discrete state representation. These representations

lie in a finite embedding space so that goal completion can be asserted when the current embedding $\varphi(s)$ equals the goal embedding z_g. Another way to express sets of constraints is via language-based predicates. A sentence describes the constraints expressed by the goal and the state or trajectory either verifies them, or does not Hermann et al. (2017); Chan et al. (2019); Jiang et al. (2019a); Bahdanau et al. (2019a,c); Hill et al. (2020a); Cideron et al. (2020c); Colas et al. (2020b); Lynch & Sermanet (2020), see Luketina et al. (2019) for a recent review. Language can easily characterize *generic goals* such as "*grow any blue object*" Colas et al. (2020b), *relational goals* like "*sort objects by size*" Jiang et al. (2019a), "*put the cylinder in the drawer*" Lynch & Sermanet (2020) or even *sequential goals* "*Open the yellow door after you open a purple door*" Chevalier-Boisvert et al. (2019a). When goals can be expressed by language sentences, goal embeddings z_g are usually language embeddings learned jointly with either the policy or the reward function. Note that, although RL goals always express constraints on the state, we can imagine *time-extended goals* where constraints are expressed on the trajectory (see a discussion in Section 9.1.1).

Reward function. The reward function of a binary problem can be viewed as a binary classifier that evaluates whether state s (or trajectory τ) verifies the constraints expressed by the goal semantics (positive reward) or not (null reward). This binary classification setting has directly been implemented as a way to learn language-based goal-conditioned reward functions $R_g(s \mid z_g)$ in Bahdanau et al. (2019a) and Colas et al. (2020b). Alternatively, the setup described in Colas et al. (2020a) proposes to turn binary problems expressed by language-based goals into goals as specific target features. To this end, they train a language-conditioned goal generator that produces specific target features verifying constraints expressed by the binary problem. As a result, this setup can use a distance-based metric to evaluate the fulfillment of a binary goal.

Goals as a Multi-Objective Balance

Some goals can be expressed, not as desired regions of the state or trajectory space but as more general objectives that the agent should maximize. In that case, goals can parameterize a particular mixture of multiple objectives that the agent should maximize.

Goal embeddings. Here, goal embeddings are simply sets of weights balancing the different objectives $z_g = (\beta_i)_{i=[1..N]}$ where β_i is the weights applied to objective i and N is the number of objectives. Note that, when $\beta_j = 1$ and $\beta_i = v0$, $\forall i \neq j$, the agent can decide to pursue any of the objective alone. In *Never Give Up*, for example, RL agents are trained to maximize a mixture of extrinsic and intrinsic rewards Badia et al. (2020b). The agent can select the mixing parameter β that can be viewed as a goal. Building on this approach, AGENT$_{57}$ adds a control of the discount factor, effectively controlling the rate at which rewards are discounted as time goes by Badia et al. (2020a).

Reward function. When goals are represented as a balance between multiple objectives, the associated reward function cannot be represented neither as a distance metric, nor as a binary classifier. Instead, the agent needs to maximize a convex combination of the objectives: $R_g(s) = \sum_{i=1}^{N} \beta_g^i R^i(s)$ where R^i is the i^{th} of N objectives and $z_g = \beta = \beta_i^g \mid_{i \in [1..N]}$ is the set of weights.

Conclusion

This section presented a diversity of goal representations, corresponding to a diversity of reward functions architectures. However, we believe this represents only a small fraction of the diversity of goal types that humans pursue. Section 9.1 discusses other goal representations that RL algorithms could target.

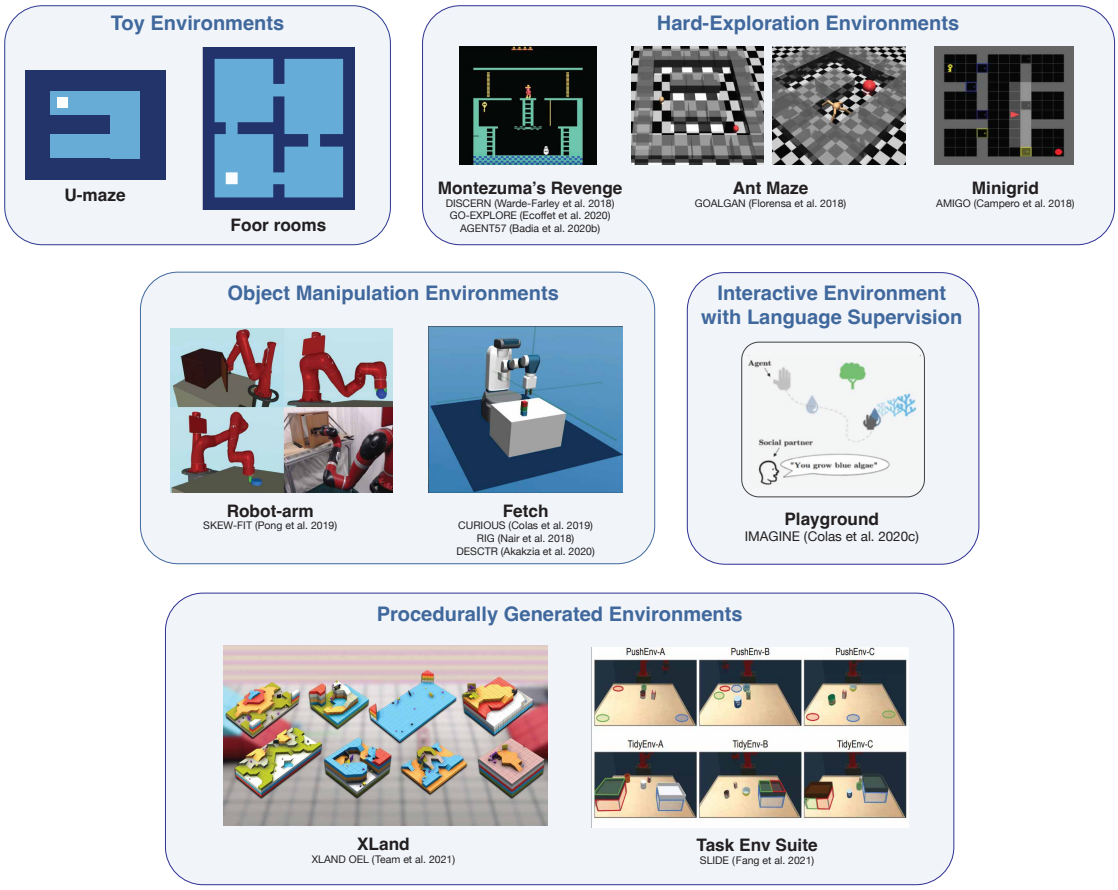


Figure 6.3: **Examples of environments in autotelic RL approaches.** We organize them by dominant feature but they might share features from other catagories as well. *Toy Envs.* are used to investigate and visualise goal-as-state coverage over 2D worlds; *Hard-Exploration Envs.* are used to benchmark goal generation algorithms; *Object Manipulation Envs.* allow for the study of the diversity of learned goals as well as curriculum learning; *Interactive Envs* permit to represent goals using language and to model interaction with caregivers; *Procedurally Generated Envs.* enhance the vastness of potentially reachable goals.

6.2.4 How to Learn Goal Representations?

The previous section discussed various types of goal representations. Autotelic agents actually need to learn these goal representations. While individual goals are represented by their embeddings and associated reward functions, representing multiple goals also requires the representation of the *support* of the goal space, i.e. how to represent the

collection of *valid goals* that the agent can sample from, see Figure 6.2. This section reviews different approaches from the literature.

Assuming Pre-Defined Goal Representation

Most approaches tackle the multi-goal RL problem, where goal spaces and associated rewards are pre-defined by the engineer and are part of the task definition. Navigation and manipulation tasks, for example, pre-define goal spaces (e.g. target agent position and target block positions respectively) and use the Euclidean distance to compute rewards Schaul et al. (2015a); Andrychowicz et al. (2017a); Nair et al. (2018a); Plappert et al. (2018); Florensa et al. (2018); Colas et al. (2019a); Blaes et al. (2019); Lanier et al. (2019); Ding et al. (2019); Li et al. (2020). Akakzia et al. (2021a); Ecoffet et al. (2021) hand-define abstract state representation and provide positive rewards when these match target goal representations. Finally, Stooke et al. (2021) hand-define a large combinatorial goal space, where goals are Boolean formulas of predicates such as *being near, on, seeing,* and *holding*, as well as their negations, with arguments taken as entities such as *objects, players*, and *floors* in procedurally-generated multi-player worlds. In all these works, goals can only be sampled from a pre-defined bounded space. This falls short of solving the intrinsically motivated skills acquisition problem. The next sub-section investigates how goal representations can be learned.

Learning Goal Embeddings

Some approaches assume the pre-existence of a goal-conditioned reward function, but learn to represent goals by learning goal embeddings. This is the case of language-based approaches, which receive rewards from the environment (thus are RL-EMGEP), but learn goal embeddings jointly with the policy during policy learning Hermann et al. (2017); Chan et al. (2019); Jiang et al. (2019a); Bahdanau et al. (2019c); Hill et al. (2020a); Cideron et al. (2020c); Lynch & Sermanet (2020). When goals are target images, goal embeddings can be learned via generative models of states, assuming the reward to be a fixed distance metric computed in the embedding space Nair et al. (2018b); Florensa et al. (2019); Pong et al. (2020); Nair et al. (2020).

Learning the Reward Function

A few approaches go even further and learn their own goal-conditioned reward function. Bahdanau et al. (2019a); Colas et al. (2020b) learn language-conditioned reward functions from an expert dataset or from language descriptions of autonomous exploratory trajectories respectively. However, the AGILE approach from Bahdanau et al. (2019a) does not generate its own goals.

In the domain of image-based goals, Venkattaramanujam et al. (2019); Hartikainen et al. (2020) learn a distance metric estimating the square root of the number of steps required to move from any state s_1 to any s_2 and generates internal signals to reward agents for getting closer to their target goals. Warde-Farley et al. (2019) learn a similarity

metric in the space of controllable aspects of the environment that is based on a mutual information objective between the state and the goal state s_g.

Wu et al. (2019) compute a distance metric representing the ability of the agent to reach one state from another using the Laplacian of the transition dynamics graph, where nodes are states and edges are actions. More precisely, they use the eigenvectors of the Laplacian matrix of the graph given by the states of the environment as basis to compute the L2 distance towards a goal configuration.

Another way to learn reward function and their associated skills is via *empowerment* methods Mohamed & Rezende (2015); Gregor et al. (2016); Achiam et al. (2018); Eysenbach et al. (2019); Dai et al. (2020); Sharma et al. (2020); Choi et al. (2021). Empowerment methods aim at maximizing the mutual information between the agent's actions or goals and its experienced states. Recent methods train agents to develop a set of skills leading to maximally different areas of the state space. Agents are rewarded for experiencing states that are easy to discriminate, while a discriminator is trained to better infer the skill z_g from the visited states. This discriminator acts as a skill-specific reward function.

All these methods set their own goals and learn their own goal-conditioned reward function. For these reasons, they can be considered as complete autotelic RL algorithms.

Learning the Support of the Goal Distribution

The previous sections reviewed several approaches to learn goal embeddings and reward functions. To represent collections of goals, one also needs to represent the support of the goal distribution — which embeddings correspond to valid goals and which do not.

Most approaches consider a pre-defined, bounded goal space in which any point is a valid goal (e.g. target positions within the boundaries of a maze, target block positions within the gripper's reach) Schaul et al. (2015a); Andrychowicz et al. (2017a); Nair et al. (2018a); Plappert et al. (2018); Colas et al. (2019a); Blaes et al. (2019); Lanier et al. (2019); Ding et al. (2019); Li et al. (2020). However, not all approaches assume pre-defined goal spaces.

The *option framework* Sutton et al. (1999); Precup (2000b) proposes to train a high-level policy to compose sequences of behaviors originating from learned low-level policies called *options*. Each option can be seen as a goal-directed policy where the goal embedding is represented by its index in the set of options. When options are policies aiming at specific states, *option discovery* methods learn the support of the goal space; they learn which goal–state are most useful to organize higher-level behaviors. *Bottleneck states* are often targeted as good sub-goals. McGovern & Barto (2001) propose to detect states that are common to multiple successful trajectories. Simsek & Barto (2004) propose to select state with maximal relative novelty, i.e. when the average novelty of following states is higher than the average novelty of previous ones. Simsek & Barto (2008) propose to leverage measures from graph theory.

The option-critic framework then opened the way to a wealth of new approaches Bacon et al. (2017). Among those, methods based on *successor features* Barreto et al. (2017,

2020); Ramesh et al. (2019) propose to learn the option space using reward embeddings. With successor features, the Q-value of a goal can be expressed as a linear combination of learned reward features, efficiently decoupling the rewards from the environmental dynamics. In a multi-goal setting, these methods pair each goal with a reward embedding and use *generalized policy improvement* to train a set of policies that efficiently share relevant reward features across goals. These methods provide key mechanisms to learn to discover and represent sub-goals. However, they do not belong to the RL-IMGEP family since high-level goals are externally provided.

Some approaches use the set of previously experienced representations to form the support of the goal distribution Veeriah et al. (2018); Akakzia et al. (2021a); Ecoffet et al. (2021). In Florensa et al. (2018), a Generative Adversarial Network (GAN) is trained on past representations of states ($\varphi(s)$) to model a distribution of goals and thus its support. In the same vein, approaches handling image-based goals usually train a generative model of image states based on Variational Auto-Encoders (VAE) to model goal distributions and support Nair et al. (2018b); Pong et al. (2020); Nair et al. (2020). In both cases, valid goals are the one generated by the generative model.

We saw that the support of valid goals can be pre-defined, a simple set of past representations or approximated by a generative model trained on these. In all cases, the agent can only sample goals *within* the convex hull of previously encountered goals (in representation space). We say that goals are *within* training distribution. This drastically limits exploration and the discovery of new behaviors.

Children, on the other hand, can imagine creative goals. Pursuing these goals is thought to be the main driver of exploratory play in children Chu & Schulz (2020). This is made possible by the compositionality of language, where sentences can easily be combined to generate new ones. The IMAGINE algorithm leverages the creative power of language to generate such *out-of-distribution* goals Colas et al. (2020b). The support of valid goals is extended to any combination of language-based goals experienced during training. They show that this mechanism augments the generalization and exploration abilities of learning agents.

In Section 6.2.5, we discuss how agents can learn to adapt the goal sampling distribution to maximize the learning progress of the agent.

Conclusion

This section presented how previous approaches tackled the problem of learning goal representations. While most approaches rely on pre-defined goal embeddings and/or reward functions, some approaches proposed to learn internal reward functions and goal embeddings jointly.

6.2.5 How to Prioritize Goal Selection?

Autotelic agents also need to select their own goals. While goals can be generated by uninformed sampling of the goal space, agents can benefit from mechanisms optimizing goal selection. In practice, this boils down to the automatic adaptation of the goal sampling distribution as a function of the agent performance.

Automatic Curriculum Learning for Goal Selection

In real-world scenarios, goal spaces can be too large for the agent to master all goals in its lifetime. Some goals might be trivial, others impossible. Some goals might be reached by chance sometimes, although the agent cannot make any progress on them. Some goals might be reachable only after the agent mastered more basic skills. For all these reasons, it is important to endow autotelic agents learning in open-ended scenarios with the ability to optimize their goal selection mechanism. This ability is a particular case of *automatic curriculum learning* ACL applied for goal selection: mechanisms that organize goal sampling so as to maximize the long-term performance improvement (distal objective). As this objective is usually not directly differentiable, curriculum learning techniques usually rely on a proximal objective. In this section, we look at various proximal objectives used in automatic curriculum learning strategies to organize goal selection. Interested readers can refer to Portelas et al. (2020b), which present a broader review of ACL methods for RL. Note that knowledge-based IMs can rely on similar proxies but focus on the optimization of the experienced states instead of on the selection of goals (e.g. maximize next-state prediction errors). A recent review of knowledge-based IM approaches can be found in Linke et al. (2020).

Intermediate or uniform difficulty.

Intermediate difficulty has been used as a proxy for long-term performance improvement, following the intuition that focusing on goals of intermediate difficulty results in short-term learning progress that will eventually turn into long-term performance increase. GOALGAN assigns feasibility scores to goals as the proportion of time the agents successfully reaches it Florensa et al. (2018). Based on this data, a GAN is trained to generate goals of intermediate difficulty, whose feasibility scores are contained within an intermediate range. Sukhbaatar et al. (2018) and Campero et al. (2021) train a goal policy with RL to propose challenging goals to the RL agent. The goal policy is rewarded for setting goals that are neither too easy nor impossible. In the same spirit, Stooke et al. (2021) use a mixture of three criteria to filter valid goals: 1) the agent has a low probability of scoring high; 2) the agent has a high probability of scoring higher than a control policy; 3) the control policy performs poorly. Finally, Zhang et al. (2020) select goals that maximize the disagreement in an ensemble of value functions. Value functions agree when the goals are too easy (the agent is always successful) or too hard (the agent always fails) but disagree for goals of intermediate difficulty.

Racanière et al. (2019) propose a variant of the GOALGAN approach and train a goal generator to sample goals of all levels of difficulty, uniformly. This approach seems to lead to better stability and improved performance on more complex tasks compared to GOALGAN Florensa et al. (2018).

Note that measures of intermediate difficulty are sensitive to the presence of stochasticity in the environment. Indeed, goals of intermediate difficulty can be detected as such either because the agent has not yet mastered them, or because the environment makes them impossible to achieve sometimes. In the second case, the agent should not focus on them, because it cannot learn anything new. Estimating medium-term learning progress helps overcoming this problem (see below).

Novelty - diversity.

Warde-Farley et al. (2019); Pong et al. (2020); Pitis et al. (2020) all bias the selection of goals towards sparse areas of the goal space. For this purpose, they train density models in the goal space. While Warde-Farley et al. (2019); Pong et al. (2020) aim at a uniform coverage of the goal space (diversity), Pitis et al. (2020) skew the distribution of selected goals even more, effectively maximizing novelty. Kovač et al. (2020) proposed to enhance these methods with a goal sampling prior focusing goal selection towards controllable areas of the goal space. Finally, Fang et al. (2021) use procedural content generation (PCG) to train a task generator that produces diverse environments in which agents can explore customized skills.

These algorithms have strong connections with empowerment methods Mohamed & Rezende (2015); Gregor et al. (2016); Achiam et al. (2018); Eysenbach et al. (2019); Campos et al. (2020); Sharma et al. (2020); Choi et al. (2021). Indeed, the mutual information between goals and states that empowerment methods aim to maximize can be rewritten as:

$$I(Z, S) = H(Z) - H(Z \mid S).$$

Thus, maximizing empowerment can be seen as maximizing the entropy of the goal distribution while minimizing the entropy of goals given experienced states. Algorithm that both learn to sample diverse goals ($H(Z)$ $\nearrow$) and learn to represent goals with variational auto-encoders ($H(Z|S)$ $\searrow$) can be seen as maximizing empowerment. The recent wealth of *empowerment* methods, however, rarely discusses the link with autotelic agents: they do not mention the notion of goals or goal-conditioned reward functions and do not discuss the problem of goal representations Gregor et al. (2016); Achiam et al. (2018); Eysenbach et al. (2019); Campos et al. (2020); Sharma et al. (2020). In a recent paper, Choi et al. (2021) investigated these links and formalized a continuum of methods from empowerment to visual goal-conditioned approaches.

While *novelty* refers to the *originality* of a reached outcome, *diversity* is a term that can only be applied to a collection of these outcomes. An outcome will be said novel if it is semantically different from what exists in the set of known outcomes. A set of outcomes will be said *diverse* when outcomes are far from each other and *cover well* the space of possible outcomes. Note that agents can also express diversity in their behavior towards a unique outcome, a skill known as *versatility* Hausman et al. (2018); Kumar et al. (2020); Osa et al. (2021); Celik et al. (2021).

Medium-term learning progress.

The idea of using learning progress (LP) as a intrinsic motivation for artificial agents dates back to the 1990s Schmidhuber (1991a,b); Kaplan & Oudeyer (2004); Oudeyer et al. (2007). At that time, however, it was used as a knowledge-based IM and rewarded progress in predictions. From 2007, Oudeyer & Kaplan (2007) suggested to use it as a competence-based IM to reward progress in competence instead. In such approaches, agents estimate their LP in different regions of the goal space and bias goal sampling towards areas of high absolute learning progress using bandit algorithms Baranes & Oudeyer (2013); Moulin-Frier et al. (2014); Forestier & Oudeyer (2016); Fournier et al.

(2018, 2021); Colas et al. (2019a); Blaes et al. (2019); Portelas et al. (2020a); Akakzia et al. (2021a). Such estimations attempts to disambiguate the incompetency or uncertainty the agent could resolve with more practice (epistemic) from the one it could not (aleatoric). Agents should indeed focus on goals towards which they can make progress and avoid goals that are either too easy, currently too hard, or impossible.

Forestier & Oudeyer (2016); Colas et al. (2019a); Blaes et al. (2019) and Akakzia et al. (2021a) organize goals into modules and compute average LP measures over modules. Fournier et al. (2018) defines goals as a discrete set of precision requirements in a reaching task and computes LP for each requirement value. The use of absolute LP enables agents to focus back on goals for which performance decreases (due to perturbations or forgetting). Akakzia et al. (2021a) introduces the success rate in the value optimized by the bandit: $v = (1 - \text{SR}) \times \text{LP}$, so that agents favor goals with high absolute LP and low competence.

Hierarchical Reinforcement Learning for Goal Sequencing.

Hierarchical reinforcement learning (HRL) can be used to guide the sequencing of goals Dayan & Hinton (1993a); Sutton et al. (1998, 1999); Precup (2000a). In HRL, a high-level policy is trained via RL or planning to generate sequence of goals for a lower level policy so as to maximize a higher-level reward. This allows to decompose tasks with long-term dependencies into simpler sub-tasks. Low-level policies are implemented by traditional goal-conditioned RL algorithms Levy et al. (2018); Röder et al. (2020) and can be trained independently from the high-level policy Kulkarni et al. (2016); Frans et al. (2018) or jointly Levy et al. (2018); Nachum et al. (2018); Röder et al. (2020). In the option framework, option can be seen as goal-directed policies that the high-level policy can choose from Sutton et al. (1999); Precup (2000b). In that case, goal embeddings are simple indicators. Most approaches consider hand-defined spaces for the sub-goals (e.g. positions in a maze). Recent approaches propose to use the state space directly Nachum et al. (2018) or to learn the sub-goal space (e.g. Vezhnevets et al. (2017), or with generative model of image states in Nasiriany et al. (2019)).

6.2.6 Discussion & Conclusion

This paper defined the intrinsically motivated skills acquisition problem and proposed to view autotelic RL algorithms or RL-IMGEP as computational tools to tackle it. These methods belong to the new field of *developmental reinforcement learning*, the intersection of the developmental robotics and RL fields. We reviewed current goal-conditioned RL approaches under the lens of autotelic agents that learn to represent and generate their own goals in addition of learning to achieve them.

We propose a new general definition of the *goal* construct: a pair of compact goal representation and an associated goal-achievement function. Interestingly, this viewpoint allowed us to categorize some RL approaches as goal-conditioned, even though the original papers did not explicitly acknowledge it. For instance, we view the Never Give Up Badia et al. (2020b) and Agent 57 Badia et al. (2020a) architectures as goal-conditioned, because agents actively select parameters affecting the task at hand (parameter mixing extrinsic

and intrinsic objectives, discount factor) and see their behavior affected by this choice (goal-conditioned policies).

This point of view also offers a direction for future research. Autotelic agents need to learn to represent goals and to measure goal achievement. Future research could extend the diversity of considered goal representations, investigate novel reward function architectures and inductive biases to allow time-extended goals, goal composition and to improve generalization.

The general vision we convey in this paper builds on the metaphor of the learning agent as a curious scientist. A scientist that would formulate hypotheses about the world and explore it to find out whether they are true. A scientist that would ask questions, and setup intermediate goals to explore the world and find answers. A scientist that would set challenges to itself to learn about the world, to discover new ways to interact with it and to grow its collection of skills and knowledge. Such a scientist could decide of its own agenda. It would not need to be instructed and could be guided only by its curiosity, by its desire to discover new information and to master new skills. Autotelic agents should nonetheless be immersed in complex socio-cultural environment, just like humans are. In contact with humans, they could learn to represent goals that humans and society care about.

Approach	Goal Type	Goal Rep.	Reward Function	Goal sampling strategy
RL-IMGEPs that assume goal embeddings and reward functions				
Fournier et al. (2018)	Target features (+tolerance)	Pre-def	Pre-def	LP-Based
HAC Levy et al. (2018)	Target features	Pre-def	Pre-def	HRL
HIRO Nachum et al. (2018)	Target features	Pre-def	Pre-def	HRL
CURIOUS Colas et al. (2019a)	Target features	Pre-def	Pre-def	LP-based
CLIC Fournier et al. (2021)	Target features	Pre-def	Pre-def	LP-based
CWYC Blaes et al. (2019)	Target features	Pre-def	Pre-def	LP-based + surprise
GO-EXPLORE Ecoffet et al. (2021)	Target features	Pre-def	Pre-def	Novelty
NGU Badia et al. (2020b)	Objectives balance	Pre-def	Pre-def	Uniform
AGENT 57 Badia et al. (2020a)	Objectives balance	Pre-def	Pre-def	Meta-learned
DECSTR Akakzia et al. (2021a)	Binary problem	Pre-def	Pre-def	LP-based
SLIDE Fang et al. (2021)	Skill index	Pre-def	Pre-def	Novelty (PCG)
XLAND OEL Stooke et al. (2021)	Binary problem	Pre-def	Pre-def	Intermediate difficulty
RL-IMGEPs that learn their goal embedding and assume reward functions				
RIG Nair et al. (2018b)	Target features (images)	Learned (VAE)	Pre-def	From VAE prior
GOALGAN Florensa et al. (2018)	Target features	Pre-def + GAN	Pre-def	Intermediate difficulty
Florensa et al. (2019)	Target features (images)	Learned (VAE)	Pre-def	From VAE prior
SKEW-FIT Pong et al. (2020)	Target features (images)	Learned (VAE)	Pre-def	Diversity
SETTER-SOLVER Racanière et al. (2019)	Target features (images)	Learned (Gen. model)	Pre-def	Uniform difficulty
MEGA Pitis et al. (2020)	Target features (images)	Learned (VAE)	Pre-def	Novelty
CC-RIG Nair et al. (2020)	Target features (images)	Learned (VAE)	Pre-def	From VAE prior
AMIGO Campero et al. (2021)	Target features (images)	Learned (with policy)	Pre-def	Adversarial
GRIMGEP Kovač et al. (2020)	Target features (images)	Learned (with policy)	Pre-def	Diversity and ALP
Full RL-IMGEPs				
DISCERN Warde-Farley et al. (2019)	Target features (images)	Learned (with policy)	Learned (similarity)	Diversity
DIAYN Eysenbach et al. (2019)	Discrete skills	Learned (with policy)	Learned (discriminability)	Uniform
Hartikainen et al. (2020)	Target features (images)	Learned (with policy)	Learned (distance)	Intermediate difficulty
Venkattaramanujam et al. (2019)	Target features (images)	Learned (with policy)	Learned (distance)	Intermediate difficulty
IMAGINE Colas et al. (2020b)	Binary problem (language)	Learned (with reward)	Learned	Uniform + Diversity
VGCRL Choi et al. (2021)	Target features	Learned	Learned	Empowerment

Table 6.1: **A classification of autotelic RL-IMGEP approaches.** Autotelic approaches require agents to sample their own goals. The proposed classification groups algorithms depending on their degree of autonomy: 1) RL-IMGEPs that rely on pre-defined goal representations (embeddings and reward functions); 2) RL-IMGEPs that rely on pre-defined reward functions but learn goal embeddings and 3) RL-IMGEPs that learn complete goal representations (embeddings and reward functions). For each algorithm, we report the type of goals being pursued (see Section 6.2.3), whether goal embeddings are learned (Section 6.2.4), whether reward functions are learned (Section 6.2.4) and how goals are sampled (Section 6.2.5). We mark in bold algorithms that use a developmental approaches and explicitly pursue the intrinsically motivated skills acquisition problem.

6.3 From Autotelic to Vygotkian Agents

Abstract

Building autonomous agents able to grow open-ended repertoires of skills across their lives is a fundamental goal of artificial intelligence (AI). A promising developmental approach recommends the design of intrinsically motivated agents that learn new skills by generating and pursuing their own goals — *autotelic agents*. But despite recent progress, existing algorithms still show serious limitations in terms of goal diversity, exploration, generalization or skill composition. This perspective calls for the immersion of autotelic agents into *rich socio-cultural worlds*, an immensely important attribute of our environment that shapes human cognition but is mostly omitted in modern AI. Inspired by the seminal work of Vygotsky, we propose *Vygotskian autotelic agents* — agents able to internalize their interactions with others and turn them into *cognitive tools*. We focus on language and show how its structure and informational content may support the development of new cognitive functions in artificial agents like it does in humans. We justify the approach by uncovering several examples of new artificial cognitive functions emerging from interactions between language and embodiment in recent works at the intersection of deep reinforcement learning and natural language processing. Looking forward, we highlight future opportunities and challenges for a Vygotskian Autotelic AI research, including the use of language models as cultural models supporting artificial cognitive development.

Introduction

Humans are remarkable examples of lifelong open-ended learners. They learn to recognize objects and crawl as infants, learn to ask questions and interact with peers as toddlers, learn to master engineering, science, or arts as adults. A fundamental goal of artificial intelligence (AI) is to build autonomous agents capable of growing such open-ended repertoires of skills.

Reinforcement learning (RL) offers a mathematical framework to formalize and tackle skill learning problems. For an embodied and situated RL agent, *learning a skill* (e.g. playing chess) is about learning to act so as to maximize future *rewards* measuring progression in that skill (e.g. +1 for winning a game, -1 for losing it).Sutton & Barto (1998) Extensions based on modern deep learning methods (deep RL) have recently made the headlines by solving a wealth of problems we thought only humans could solve: beating chess and go world champions,Silver et al. (2016) controlling stratospheric balloons,Bellemare et al. (2020) or even maintaining plasma in fusion reactors.Degrave et al. (2022) But human chess world champions can also run, cook, draw a cat, or make a friend laugh. Humans are proficient in a wide diversity of tasks, most of which they just invent for themselves. The RL framework, in its standard form, considers a single predefined reward function and, thus, must be extended (see Figure 6.4, a).

Jean Piaget, a pioneer of developmental psychology, demonstrated children's ability to set their own goals, shape their own learning trajectories, and develop in symbiosis

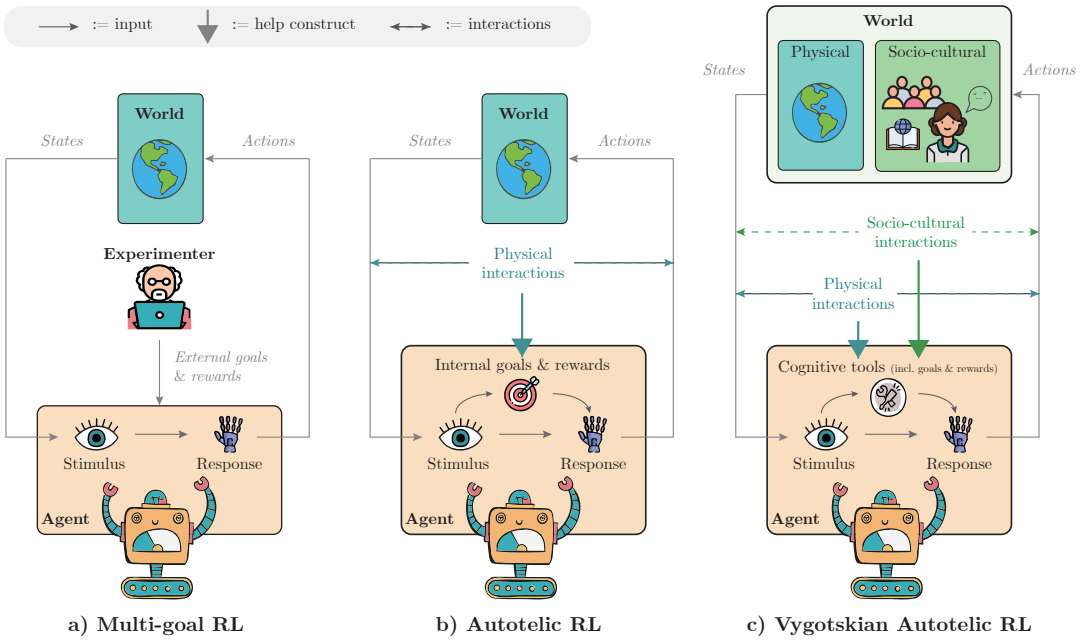


Figure 6.4: From multi-goal RL to autotelic RL to Vygotskian autotelic RL. RL defines an agent experiencing the state of the world as stimuli and acting on that world via actions. Multi-goal RL (a): goals and associated rewards come from pre-engineered functions and are perceived as sensory stimuli by the agent. Autotelic RL (b): agents build internal goal representations from interactions between their intrinsic motivations and their physical experience of the world (Piagetian view). Vygotskian autotelic RL (c): agents internalize physical and socio-cultural interactions into *cognitive tools*. Here, *cognitive tools* refer to any self-generated representation that mediates stimulus and actions. This can include self-generated goals, explanations, descriptions, attentional biases, visual aids, mnemotechnic tricks, etc.

with their physical environment.Piaget (1952) His work influenced future developments in both psychology and AI.Dautenhahn & Billard (1999) Indeed, if each skill is the association of a goal (e.g."be a good chess player") and a policy to reach it, then *open-ended skill discovery* presupposes the ability to invent and select one's own goals and reward functions so as to progressively build repertoires of skills. *Autotelic RL* — from the Greek *auto* (self) and *telos* (goal) — extends the RL framework to build such agents.Colas et al. (2022) It integrates two extensions of the standard RL framework: the ability to consider multiple goals in parallel (*multi-goal RL*) and the ability to represent and select one's own goals. Although the first extension is straightforward,Schaul et al. (2015b) the second requires another ingredient inspired by the study of human learning: *intrinsic motivations*.

Most of the human time is spent on activities that do not seem to satisfy any utilitarian end; think about children playing, or adults watching movies. Psychologists argue that such exploratory behaviors are powered by *intrinsic motivations* (IM), a set of brain processes driving us to experience situations for the mere pleasure of experiencing novelty, surprise, or learning progress.Berlyne (1966b); Kidd & Hayden (2015b); Gottlieb & Oudeyer (2018b) Similar processes can be coded into artificial agents to foster spontaneous exploratory capabilities.Schmidhuber (1991d); Barto & Simsek (2005);

Oudeyer et al. (2007) Among them, *knowledge-based* IM drive agents to experience parts of the environment to improve their internal models of the world and *competence-based* IM let them improve their mastery of self-generated goals.Oudeyer & Kaplan (2009) In the Piagetian tradition, autotelic agents see their goal representation and goal selection mechanisms emerge from interactions between competence-based IM and their experience of the physical world (see Figure 6.4, b).Colas et al. (2022)

In practice, current autotelic RL implementations still lack human-like open-endedness. The goal representations emerging from their intrinsically motivated experience with the physical world end up very concrete and mostly consist in reaching target stimuli (e.g. matching their visual input with a particular target).Colas et al. (2022) This contrasts with the wide diversity and the abstraction of goals targeted by humans. In addition, the generated goals very often belong to the distribution of previously experienced effects, which drastically limits the ability of autotelic agents to represent *creative goals*, thus to explore and undergo an open-ended discovery process.Colas et al. (2020c) Besides goal imagination, RL algorithms still lack human-like capacities in terms of generalization, skill composition, abstraction, or sample efficiency.Witty et al. (2021); Shanahan & Mitchell (2022)

The way forward might build on an alternative view of child development proposed by another pioneer of developmental psychology called Lev Vygotsky. Humans are social beings; intrinsically motivated to interact and cooperate with their peers.Tomasello (1999b); Tomasello et al. (2005); Brewer et al. (2014) For Vygotsky, linguistic social interactions such as descriptions, explanations, corrections, or play start as interpersonal processes before they are turned into *intrapersonal* cognitive processes through the process of *internalization*.Vygotsky (1934) Following his vision, many psychologists,Berk (1994); Lupyan (2012); Gentner & Hoyos (2017) linguists,Whorf (1956); Rumelhart et al. (1986); Lakoff & Johnson (2008) and philosophers Hesse (1988); Dennett (1993); Clark (1998a); Carruthers (2002) argued for the importance of socio-cultural interactions in the development of human intelligence.

What does this mean for AI? We advocate for a *Vygotskian Autotelic AI*. Specifically, we propose to immerse autotelic agents into our rich socio-cultural world; to let them interact with us and with their peers in natural language; to let them internalize these interactions and mesh them with their cognitive development (see Figure 6.4, c). Just like they do for humans, language and culture will help shape the agents' goal representations and generation mechanisms, thereby offering them the ability to generate more diverse and abstract goals; to imagine new goals beyond their past experience. Because they will develop at our contact, bathed in our cultures, they will learn about our cultural norms, values, customs, interests, and thought processes; all of which would be impossible to learn in social isolation. Just like humans, machines will use language to develop higher cognitive functions like abstraction, generalization, or imagination.Carruthers & Boucher (1998); Gentner & Hoyos (2017); Dove (2018) As we will see, this process has already started.

Recent advances in AI make our proposition particularly timely. Indeed, these past years have seen a revolution in natural language processing (NLP), the set of tools designed to analyze and generate language. Generative models of language trained on gigantic amounts of text can now produce high-quality language,Brown et al. (2020)

handle multimodal inputs,Radford et al. (2021); Ramesh et al. (2022); Alayrac et al. (2022) and capture common senseWest et al. (2022) or cultural artefacts.Hershcovich et al. (2022); Arora et al. (2022) Pure language models like GPT-3Brown et al. (2020) are capable of impressive zero-shot generalizations including joke explanations, arithmetic, question answering, or even multi-step reasoning when nudged appropriately.Creswell et al. (2022) Multi-modal variants can explain visual jokes (memes)Alayrac et al. (2022) or generate impressive images from the most creative descriptions its testers can come up with.Ramesh et al. (2021, 2022) The ongoing convergence of autotelic RL and NLP will offer a wealth of opportunities as autotelic agents learn to interact with us, learn from us and teach us back. This motivates the elaboration of a theoretical framework to understand recent linguistic RL developments and point towards future challenges in the quest for open-ended skill discovery.

This perspective extends previous calls to leverage Vygotsky's insights for a more socially-situated cognitive robotics.Dautenhahn & Billard (1999); Zlatev (2001); Lindblom & Ziemke (2003); Mirolli & Parisi (2011) Zlatev discussed interactions between social-situatedness and epigenetic development,Zlatev (2001) Dautenhahn and Billard drew the parallel between AI and the Piagetian vs. Vygotskian views,Dautenhahn & Billard (1999) while Mirolli and Parisi, as well as Cangelosi et al., reviewed the first successful auxiliary uses of language for decision-making.Mirolli & Parisi (2011); Cangelosi et al. (2010a) In the last decade however, the AI community seems to have lost track of these insights. Today we update these arguments in the light of recent AI advances and reframe the Vygotskian perspective within the autotelic RL framework. As a result, we will not discuss non-embodied multi-modal supervised techniquesRadford et al. (2021); Ramesh et al. (2022); Alayrac et al. (2022) or non-linguistic autotelic RL.Schaul et al. (2015b) We will not cover the advances in the sub-field of RL dedicated to the computational modeling of social interactions (social RL).Jaques et al. (2019) Although future Vygotskian autotelic agents must incorporate social RL mechanisms, current approaches do not consider autotelic agents able to set their own goals and, as such, do not tackle the open-ended skill discovery problem (see a complete discussion in a related paperSigaud et al. (2021b)).

The next section sets the background and discusses the interaction between language and thought in humans by building on the work of psychologists and philosophers (Section 6.3.1). The two following sections dive into the two key elements of a Vygotskian autotelic AI identified in Section 6.3.1: 1) the ability to exploit information contained in linguistic structure and content (syntax, vocabulary, narratives) to support the development of cognitive functions (Section 6.3.2); 2) the ability to internalize linguistic interactions within the agent to power its future autonomy and integration into the sociocultural world (Section 6.3.3). Finally, Section 9.2 identifies open challenges for future research.

6.3.1 Language and Thought in Humans, a Vygotskian Perspective

Our ability to generate new ideas is the source of our incredible success in the animal kingdom. But this ability did not appear with the first *homo sapiens* 130,000 years ago. Indeed, the oldest imaginative artifacts such as figurative arts, elaborate burials or the first dwellings only date back to 70,000 years ago.Harari (2014); Vyshedskiy (2019) This is

thought to coincide with the apparition of *recursive language*.Goldberg (1999); Vyshedskiy (2019); Hoffmann (2020) Which of these appeared first? Creativity or recursive language? Or did they mutually bootstrap?

Extreme views on the topic either characterize language as a pure communicative device to convey our inner thoughts (strong communicative thesis)Chomsky (1957b); Fodor (1975) or, on the other hand, argue that only language can be the vehicle of our thoughts (strong cognitive thesis)Wittgenstein (1953); McDowell (1996) As often, the truth seems to lie in between. Animal and preverbal infants demonstrate complex cognition,Sperber et al. (1995); Allen & Bekoff (1999) but language does impact the way we perceive,Waxman & Markow (1995); Yoshida & Smith (2003) represent concepts,Lakoff & Johnson (2008) conduct compositional and relational thinking,Gentner & Loewenstein (2002); Gentner & Hoyos (2017); Vyshedskiy (2019) etc. Thus, language seems to be at least *required* to develop some of our cognitive processes (requirement thesis), and might still be the vehicle of *some* of our thoughts (constitutive thesis).Carruthers & Boucher (1998) Interested readers can find a thorough overview of this debate in *Language and Thought* by Carruthers and Boucher.Carruthers & Boucher (1998)

If language is required to develop some of our higher cognitive functions, then autotelic artificial agents should use it as well. But how does that work? What is so special about language? Let us start with *words*, which some called *invitations to form categories*.Waxman & Markow (1995) Hearing the same word in a variety of contexts invites humans to compare situations, find similarities, differences and build symbolic representations of agents, object and their attributes. With words, the continuous world can be simplified and structured into mental entities at various levels of abstraction.

The recursivity and partial compositionality of language allows us to readily understand the meaning of sentences we never heard before by generalizing from known words and syntactic structures. On the flip side, it also supports *linguistic productivity*,Chomsky (1957b) the ability to generate new sentences — thus new ideas — in an open-ended way. Relational structures such as comparisons and metaphors facilitate our relational thinking,Gentner & Loewenstein (2002); Gentner & Hoyos (2017) condition our ability to compose mental images,Vyshedskiy (2019) and support our understanding of abstract concepts such as emotions, politics or scientific theories.Hesse (1988); Lakoff & Johnson (2008)

Finally, language is a cultural artefact inherited from previous generations and shared with others. It supports our cultural evolution and allows humans to efficiently transfer knowledge and practices across people and generationsHenrich & McElreath (2003); Morgan et al. (2015); Chopra et al. (2019) — a process known as the *cultural rachet*.Tomasello (1999b) Through shared cultural artefacts such as narratives, we learn to share common values, customs and social norms, we learn how to navigate the world, what to attend to, how to think, what to expect from others.Bruner (1990) This cultural knowledge is readily accessible to children as they enter societies via social interactions and formal education. Learning language further extends the access to cultural artefacts such as books, movies, or the Internet. These act as a thousand virtual social partners to learn from.

We now understand why language is so special. Let us focus on how it can shape cognitive development in humans and machines. Dennett, a proponent of the requirement

thesis, suggests that linguistic exposition alone can lead to a fundamental cognitive reorganization of the human brain.Dennett (1993) He compares it to the installation of a serial virtual machine on humans' massively parallel processing brains. As a result, a slight change in our computational hardware (e.g. compared to our primates relatives) could open the possibility for any cognitive software reprogramming driven by language, in turn triggering the learning and cultural evolution of higher cognitive capacities. Carruthers, a proponent of the constitutive thesis, suggests that language may have evolved as a separate module to exchange inner representations with our peers (naive physics, theory of mind, etc). This would require connections between linguistic and non-linguistic modules to allow conversions between inner representations and linguistic inputs/outputs. In a similar way that humans can trigger imagined visual representations via top-down connections in their visual cortex, top-down activations of the linguistic module would create *inner speech*. This hallucinated speech, when broadcast to other modules, would implement *thinking in language*.Carruthers (1998) Clark advances yet another possibility, the *supra-communicative view*. Here, language does not transform the way the brain makes computations and is not the vehicle of thoughts. Instead, language complements our standard computation activities by "*re-shaping the computational spaces*," turning problems that would be out of reach into problems our pattern-matching brains can solve.Clark (1998a) In that sense, language is a *cognitive tool* that enhances our cognitive abilities without altering them per se.

Vygotsky's theory brings a complementary argument to this debate. Caretakers naturally scaffold the learning experiences of children, tailoring them to their current objectives and capacities. Through encouragement, attention guidance, explanations or plan suggestions, they provide cognitive aids to children in the form of interpersonal social processes.Vygotsky (1934) In this *zone of proximal development*, as Vygotsky coined it, children can benefit from these social interactions to achieve more than they could alone. In these moments, children *internalize* linguistic and social aids and progressively turn these interpersonal processes into intrapersonal *psychological tools*.Vygotsky (1934) This essentially consists in building internal models of social partners such that learners can self-generate contextual guidance in the absence of an external one. Social speech is internalized into private speech (an outer speech of children for themselves), which, as it develops, becomes more goal-oriented and provides cognitive aids of the type caretakers would provide.Vygotsky (1934); Berk (1994) Progressively, it becomes more efficient and abbreviated, less vocalized, until it is entirely internalized by the child and becomes *inner speech*.

This section showed why language is so important and might just be required for the development of our highest cognitive functions. If we want machines to show more human-like open-ended skill discovery processes, we might need to immerse them into rich socio-cultural worlds from the very beginning — just like we do with children — and equip them with tools to benefit from them. From the arguments above, we identify two key elements, see Figure 6.5. First, we need autotelic agents to exploit the information contained within linguistic structures and contents. Exposed to language, agents will reorganize their internal representations for better abstraction, generalization and better alignment with human values, norms and customs (Dennett's thesis). Second, we need autotelic agents to internalize social interactive processes, i.e. to *model social partners within themselves* (Vygotsky's internalization). Social processes, turned into

intrapersonal cognitive processes will orient the agent's focus, help it decompose tasks or imagine goals. This inner speech generation can serve as a common currency between other modules (e.g. perception, motor control, goal generation) in line with Carruther's view and will help agent project problems onto linguistic spaces where they might be easier to solve (Clark's view). In the next two sections (6.3.2–6.3.3), we discuss these two key elements in detail and reframe recent works at the intersection of RL and language in their light.

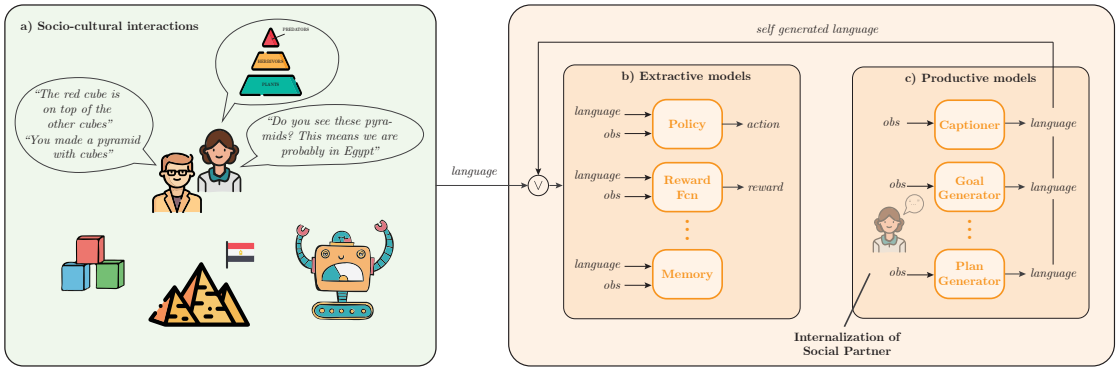


Figure 6.5: The three components of Vygotskian autotelic agents: socio-cultural interactions, linguistic extraction and internalized linguistic production. Vygotskian autotelic agents are immersed into rich socio-cultural worlds where they experience a variety of linguistic feedback including descriptions, explanations, or metaphors (a). They can exploit information from linguistic structures and content by conditioning their internal modules on this feedback (b, extractive models). Finally, they learn to internalize social interactions by training productive models of language to generate feedback similar to the one they receive from others (c, productive models). This offers agents the autonomy to build their own cognitive tools, bootstrapped by socio-cultural language.

6.3.2 Exploiting Linguistic Structure and Content

In its vocabulary, syntax and narratives, language offers both powerful computational tools for thinking and important cultural knowledge about the world. According to Dennett's thesis, mere exposure to language can already help agents rewire their inner processes and develop new abilities. Recent advances in AI seem to support that idea.

Learning to abstract and generalize. Exposition to linguistic labels is known to facilitate category learning in humans,Waxman & Markow (1995); Yoshida & Smith (2003) but also in machines.Lupyan (2005) As Mirolli and Parisi defend, the repeated occurrence of a linguistic label (*red* in their example) leads to the conflation of internal representations associated with that label (red things) which, in turn, facilitates further classifications based on the linguistic attributes.Mirolli & Parisi (2011)

We see a similar effect in RL agents targeting linguistic goals. The exposure to aligned instructions and trajectories seems to reshape the internal representations of the agent contained within its action policy. The policy is a neural network-based function

conditioned on the agent's instruction that maps the current state of the world to its next actions. By *internal representations*, we mean representations computed within the layers of the policy to facilitate the final decision-making. When repeatedly asked to grasp *red objects*, the policy learns to focus on objects' colors to facilitate action selection.Hill et al. (2020c); Colas et al. (2020c) *Red* is an abstraction over a continuous space of colors. It is first cultural, outside of the agent, but gets progressively internalized within the agent via a combination of linguistic exposure and decision-making.

Exposed to a diversity of instructions, agents gain new cognitive abilities. The first is *abstraction*. Linguistic autotelic agents can reach and make sense of abstract relational goals "*sort objects by size*,"Jiang et al. (2019b) "*put the cylinder in the drawer*,"Lynch & Sermanet (2021) sequential goals "*open the yellow door after opening a red door*,"Chevalier-Boisvert et al. (2019b) or even learning goals "*is the ghargh edible?*."Yuan et al. (2019a) Whereas handling abstract goals used to require engineers to hard-code specific goal representations and reward functions within the agent,Colas et al. (2019a); Stooke et al. (2021) linguistic goals offer abstraction via simple linguistic interactions.**?**Colas et al. (2020c) Once abstractions have been distilled within the representations of the agent, they can be leveraged to augment its exploratory capacities. Searching for novelty in a space of abstract linguistic descriptions of the world is indeed more efficient than searching for novelty in low-level sensorimotor spaces which could be trivially triggered by leaves moving in the wind or TV noise.Tam et al. (2022); Mu et al. (2022)

A second cognitive ability is *systematic generalization*. Language-instructed agents indeed seem to demonstrate the ability to generalize to new instructions obtained by systematic recombinations of instructions they were trained on.Hill et al. (2020c) For instance, agents that learned to *grasp blue objects* and *put green objects on the table* can directly *grasp green objects* and *put blue objects on the table*.Hermann et al. (2017); Chevalier-Boisvert et al. (2019b); **?**); Hill et al. (2020c,d); Colas et al. (2020c); Sharma et al. (2021); Karch et al. (2021) This ability can either be encoded in learning architecture through the use of modular networks (neuro-symbolic approaches), or emerge spontaneously in plain networks under the right environmental conditions.Hill et al. (2020c) Although sometimes the world does not conform to strict linguistic compositionality, systematic generalization still supports good priors — e.g. *feeding the cat* is not a strict transposition of *feeding the plant* but they still share similarities (bringing supplies to the cat/plant).Colas et al. (2020c)

Learning to represent possible futures. After being exposed to aligned trajectories and linguistic descriptions, agents can generate concrete examples of abstract descriptions. The DECSTR approach, for example, trains a generative world model to sample from the distribution of possible future states matching a given abstract linguistic description.Akakzia et al. (2021b) This simple mapping supports *behavioral diversity*, the ability to represent different possible futures so as to select one to pursue. Similar setups could leverage DALL-E, an impressive text-to-image generative system.Ramesh et al. (2021, 2022) Trained on pairs of images and compositional descriptions, DALL-E can generate high-quality images from the most twisted descriptions humans can think of. The exposition to compositional language, paired with sufficiently powerful learning architectures and algorithms leads to impressive visual composition abilities that could be put to use to generate visual goals or to represent possible futures in embodied and situated agents.

Learning to decompose tasks. Vygotsky and others discovered that children's use of private speech helps them increase self-control and is instrumental to their capacity to reason and solve hard tasks.Vygotsky (1934); Berk (1994) The ability to formulate sentences like "at the left of the blue wall," for instance, predicts spatial orientation capacities in such contexts, while interfering with adult's inner speech via speaking tasks hinders theirs.Hermer-Vazquez (2001)

Language indeed contains cues about how to decompose tasks into sub-tasks, i.e. how to *generate good plans*. Although *gharble* is a made-up word, *fry the gharble* probably involves a preparation of the gharble (e.g. peeling, cutting), some sort of oil and a frying pan.Yuan et al. (2019a) *Draw an octogon* contains cues about the decomposition of the task: *octo* means 8, so we should probably do something 8 times, etc.Wong et al. (2021) Recent AI approaches leverage these regularities by training *plan generators* from linguistic task descriptions.Jiang et al. (2019b); Chen et al. (2021); Sharma et al. (2021); Mirchandani et al. (2021); Shridhar et al. (2021); Wong et al. (2021) Among them, Wong et al. use plan generation as an auxiliary task to train a drawing policy.Wong et al. (2021) Generating plans to solve a particular drawing task helps shape the internal representation of the main policy which, they find, favors abstraction and generalization in the main task. Interestingly, language only shapes representations and is not required at test time, in line with the requirement thesis of Dennett.

Inspired by video games of the 80s such as *Zork*, text-based environments define purely linguistic goals, actions and states.Côté et al. (2018); Das et al. (2018a); Yuan et al. (2019a) Training a policy in such environments can be seen as training a plan generator in a linguistic world model, i.e. training an inner speech to generate good task decompositions. This idea was exploited in *AlfWorld*, where a pre-trained plan generator is deployed in a physical environment to generate sub-goals for a low-level policy.Shridhar et al. (2021) Here, the abstraction capabilities of language help the plan generator solve long-horizon tasks.

The above approaches echo the thesis of Dennett (Section 6.3.1): the mere exposure to structured language, once internalized within internal modules (reward function, policy, world model) strongly shapes inner representations in new ways and supports new cognitive functions (abstraction, future states generation, compositional generalization, task decomposition, etc).

Learning from cultural artefacts. Large language models (LLM) are trained on huge quantities of text scrapped from the internet: Wikipedia, forums, blogs, scientific articles, books, subtitles, etc.Devlin et al. (2019); Brown et al. (2020) As such, they can be seen as *cultural models* that contain information about our values, norms, customs, history, or interests.Hershcovich et al. (2022); Arora et al. (2022) This represents a great opportunity for autotelic agents to learn about us, align with us, and better navigate our complex world. So far, only very little research has leveraged that opportunity. An example is the use of a trained LLM to act as a zero-shot planner, i.e. a plan generator.Huang et al. (2022) Plugged with an interactive agent, the language model is used to generate sub-goals for the agent to solve the main task. Another work extracts information about complex time-extended behaviors from an LLM by asking it to score the actions available to the agent.Ahn et al. (2022). Finally, the MineDojo frameworkFan et al. (2022) proposes to caption thousands of YouTube videos of humans playing Minecraft

using GPT-3Brown et al. (2020), generating creative high-level tasks as well as low-level linguistic guidance for embodied agents. Challenge #3 of Section 9.2 discusses more opportunities to harness these cultural models for open-ended skill discovery.

6.3.3 Internalization of Language Production

Agents that internalize extractive models learn to exploit the information contained within linguistic vocabularies, structures and narratives. However, most of them require external linguistic inputs at test time and, thus, cannot be considered autonomous. Vygotskian autotelic agents reach autonomy by internalizing *productive models*; i.e. by learning to generate their own linguistic inputs, their own *inner speech* (see Figure 6.5, c).

Inner speech can be understood as a fully-formed language: descriptions, explanations or advice to be fed back to extractive models; to serve as a common currency between cognitive modules (fully-formed inner speech).Zeng et al. (2022) But it might also be understood as *distributed representations* within productive models, *upstream from fully-formed language* (distributed inner speech). In the latter interpretation, linguistic production acts as an auxiliary task whose true purpose is to shape the agent's cognitive representations. Symbolic behaviors might indeed not require explicit symbolic representations but may emerge from distributed architectures trained on structured tasks, e.g. involving linguistic predictions.McClelland et al. (2010); Santoro et al. (2021) In the literature, we found four types of productive models making use of either fully-formed or distributed inner speech: trajectory captioners, plan generators, explanation generators and goal generators.

Trajectory captioners. Trajectory captioners are trained on instructive or descriptive feedback to generate valid descriptions of scenes or trajectories.Cideron et al. (2020b); Zhou & Small (2020b); Colas et al. (2020c); Nguyen et al. (2021); Yan et al. (2022) In line with Vygotsky's theory, these agents internalize models of descriptive social partners. They generate an *inner speech* describing their ongoing behaviors just like a caretaker would. Used as an auxiliary task (distributed inner speech), the generation of descriptions helps the agent shape its representation so as to generalize better to new tasks.Yan et al. (2022) With fully-formed inner speech, agents can generate new multi-modal data autonomously, and learn from past experience via *hindsight learning*,Andrychowicz et al. (2017b) i.e. the reinterpretation of their trajectory as a valid behavior to achieve the trajectory's description.Zhou & Small (2020b); Colas et al. (2020c); Nguyen et al. (2021)

Plan generators. Plan generators are both extractive and productive. Following the formalism of hierarchical RL (HRL), plan generators are implemented by a *high-level policy* generating linguistic sub-goals to a low-level policy (executioner).Dayan & Hinton (1993b); Sutton et al. (1999) Linguistic sub-goals are a form of inner speech that facilitates decision-making at lower temporal resolution by providing abstract, human-interpretable actions, which themselves favor systematic generalization for the low-level policy (see Section 6.3.2).Jiang et al. (2019b); Chen et al. (2021); Shridhar et al. (2021) Here, agents internalize linguistic production to autonomously generate further guidance for themselves in fully-formed language (task decompositions).

Explanation generators. Vygotskian agents can generate *explanations*. Using the generation of explanations as an auxiliary task (distributed inner speech) was indeed shown to support causal and relational learning in complex *odd one out* tasks.Lampinen et al. (2022) Note however that this approach is neither embodied, nor autotelic.

Goal generators. Some forms of creativity appear easier in linguistic spaces because swapping words, compositing new sentences, and generating metaphors are all easier in the language space than in sensorimotor spaces. The IMAGINE approach leverages this idea to support *creative goal imagination*.Colas et al. (2020c) While previous methods were limited to generating goals within the distribution of past experience (e.g. with generative models of statesNair et al. (2018b)), IMAGINE invents out-of-distribution goals by combining descriptions of past goals. These manipulations occur in linguistic spaces directly and are thus *linguistic thoughts*; fully-formed inner speech (Carruthers' view). The problem of goal imagination, difficult to solve in sensorimotor space, is projected onto the linguistic space, solved there, and projected back to sensorimotor space (Clark's view). This, in turn, powers additional cognitive abilities. First, it powers a creative exploration oriented towards objects and interactions with them. Second, it enhances systematic generalization by widening the set of goals the agent can train on.Colas et al. (2020c)

By internalizing linguistic production, IMAGINE generates goals that are both *novel* (new sentences) and *appropriate* (they respect linguistic regularities, both structures, and contents).Runco & Jaeger (2012) Social descriptions focus on objects, object attributes, and interactions with these objects. Imagined goals obtained by recompositions of social ones share the same attentional and conceptual biases, e.g. by reusing semantic categories of a particular culture. Thus, cultural biases are implicitly transmitted to the agent, which forms goal representations and biases goal selection following cultural constraints.Colas et al. (2020c)

Note that productive models are very rare in the literature. In the future, Vygotskian autotelic agents must learn to internalize productive models for all types of multi-modal feedback they encounter: advice, explanations, attention guidance, motivation, instructions, descriptions, etc. It is only by learning to generate this guidance for themselves that they may gain full control of their own behavior. The question of whether to use fully-formed or distributed inner speech remains open, as both strategies seem to find different use-cases. Because cultural models are biased, future agents will need to edit, correct, augment and generate their own interpretations of culture based on their individual experiences. How to efficiently steer language models in these ways remains a question to explore in future research.

Conclusion

This perspective builds on the autotelic RL framework inspired by the Piagetian tradition to propose a complementary view motivated by Vygotsky's work: a *Vygotskyan Autotelic AI*. Recent works at the intersection of deep RL and NLP present promising first steps, but a lot remains to be done. Vygotskian autotelic agents will interact with us and with our culture, exploit linguistic structures and content and finally internalize social interactions to serve as the basis of their future cognitive functions: abstraction, compositional thinking, generalization or imagination.

Vygotskian Autotelic AI opens a new research program with exciting opportunities. The current NLP revolution can be harnessed to design more interactive and teachable agents. As they interact with humans and their peers in rich socio-cultural worlds, they will learn to leverage cultural and linguistic knowledge to bootstrap their cognitive development, and may eventually contribute back to our shared cultural evolution. This perspective uncovers an important challenge: how will we train and leverage cultural models to align with the values of human cultures so as to educate ethical and useful artificial agents?

Dennett used the software/hardware metaphor to describe the impact of language on human brains.Dennett (1993) Current AI research mostly focuses on *hardware* by asking *how can we design better learning architectures and algorithms?* In complement, this paper suggests to focus on *software* as well: *How can we build the right socio-cultural bath which, combined with efficient hardware, will allow the emergence of a more human-like AI*?

Chapter 7

Alignment: Grounding Spatio-Temporal Language with Transformers

Contents

Abstract

Language is an interface to the outside world. In order for embodied agents to use it, language must be grounded in other, sensorimotor modalities. While there is an extended literature studying how machines can learn grounded language, the topic of how to learn spatio-temporal linguistic concepts is still largely uncharted. To make progress in this direction, we here introduce a novel spatio-temporal language grounding task where the goal is to learn the meaning of spatio-temporal descriptions of behavioral traces of an embodied agent. This is achieved by training a truth function that predicts if a description matches a given history of observations. The descriptions involve time-extended predicates in past and present tense as well as spatio-temporal references to objects in the scene. To study the role of architectural biases in this task, we train several models including multimodal Transformer architectures; the latter implement different attention computations between words and objects across space and time. We test models on two classes of generalization: 1) generalization to randomly held-out sentences; 2) generalization to grammar primitives. We observe that maintaining object identity in the attention computation of our Transformers is instrumental to achieving good performance on generalization

overall, and that summarizing object traces in a single token has little influence on performance. We then discuss how this opens new perspectives for language-guided autonomous embodied agents. We also release our code under open-source license as well as pre-trained models and datasets to encourage the wider community to build upon and extend our work in the future.

7.1 Motivations

Building autonomous agents that learn to represent and use language is a long standing-goal in *Artificial Intelligence* Glenberg & Kaschak (2002b); Steels (2006). In developmental robotics Cangelosi et al. (2010b), language is considered a cornerstone of development that enables embodied cognitive agents to learn more efficiently through social interactions with humans or through cooperation with other autonomous agents. It is also considered essential to develop complex sensorimotor skills, facilitating the representation of behaviors and actions.

Embodied Language Grounding Zwaan & Madden (2005b) is the field that studies how agents can align language with their behaviors in order to extract the meaning of linguistic constructions. Early approaches in developmental robotics studied how various machine learning techniques, ranging from neural networks Sugita & Tani (2005); Tuci et al. (2011); Hinaut et al. (2014) to non-negative matrix factorization Mangin et al. (2015), could enable the acquisition of grounded compositional language Taniguchi et al. (2016); Tani (2016). This line of work was recently extended using techniques for *Language conditioned Deep Reinforcement Learning* Luketina et al. (2019). Among these works we can distinguish mainly three language grounding strategies. The first one consists of directly grounding language in the behavior of agents by training goal-conditioned policies satisfying linguistic instructions Sugita & Tani (2005); Tuci et al. (2011); Hill et al. (2020b); **?**); **?**. The second aims at extracting the meaning of sentences from mental simulations (i.e. generative models) of possible sensorimotor configurations matching linguistic descriptions Mangin et al. (2015); Akakzia et al. (2021c); Cideron et al. (2020a); Nguyen et al. (2021). The third strategy searches to learn the meaning of linguistic constructs in terms of outcomes that agents can observe in the environment. This is achieved by training a truth function that detects if descriptions provided by an expert match certain world configurations. This truth function can be obtained via *Inverse Reinforcement Learning* Zhou & Small (2020a); Bahdanau et al. (2019b) or by training a multi-modal binary classifier Colas et al. (2020b).

While all the above-mentioned approaches consider language that describes immediate and instantaneous actions, we argue that it is also important for agents to grasp language expressing concepts that span multiple time scales. We thus propose to study the grounding of new spatio-temporal concepts enabling agents to ground time extended predicates (Fig. 7.1a) with complex spatio-temporal references to objects (Fig. 7.1b) and understand both present and past tenses (Fig. 7.1c). To do so we choose the third strategy mentioned above, i.e. to train a truth function that predicts when descriptions match traces of experience. This choice is motivated by two important considerations. First, prior work showed that learning truth functions was key to foster generalization Bahdanau et al. (2019b), enabling agents to efficiently self-train policies via goal imagination

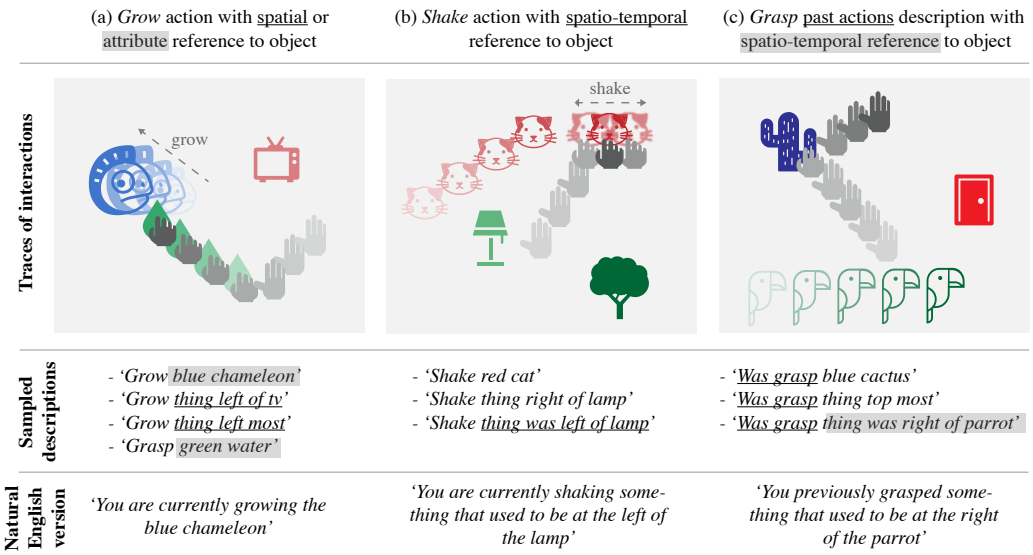


Figure 7.1: **Visual summary of the Temporal Playground environment:** At each episode (column a, b and c), the actions of an agent (represented by a hand) unfold in the environment and generate a trace of interactions between objects and the agent body. Given such a trace, the environment automatically generates a set of synthetic linguistic descriptions that are true at the end of the trace. In (a) the agent grows an object which is described with spatial (underlined) or attribute (highlighted) reference. In (b) it shakes an object which is described with attribute, spatial or spatio-temporal (underlined) reference. In (c) it has grasped an object (past action underlined) which is described with attribute, spatial or spatio-temporal (highlighted) reference.

Colas et al. (2020a) and goal relabeling Cideron et al. (2020a). Hence the truth function is an important and self-contained component of larger learning systems. Second, this strategy allows to carefully control the distribution of experiences and descriptions perceived by the agent.

Grounding spatio-temporal language is a relational problem. In the context of this paper, the concepts we aim at grounding are temporal and spatial, and thus relational by nature. But more generally, it is worth mentioning that Embodied Language Grounding has a relational structure. We understand the meaning of words by analyzing the relations they state in the world Gentner & Loewenstein (2002). Actions are relations between subjects and objects, and can be defined in terms of affordances of the agent Gibson (1968). As a result we implement our truth function using relational architectures based on *Transformers* Vaswani et al. (2017) and investigate the role of the relational bias Battaglia et al. (2018) on learning. We propose a formalism unifying three variants of a multi-modal transformer inspired by Ding et al. (2020) that implement different relational operations. We measure the generalization capabilities of these architectures along three axis 1) generalization to new traces of experience; 2) generalization to randomly held out sentences; 3) generalization to grammar primitives, systematically held out from the training set as in Ruis et al. (2020). We observe that maintaining object identity in the attention computation of our Transformers is instrumental to achieving good performance on generalization overall. We also identify specific relational operation that are key to generalize on certain grammar primitives.

Contributions.

This paper introduces:

1. A new Embodied Language Grounding task focusing on spatio-temporal language;
2. A formalism unifying different relational architectures based on Transformers expressed as a function of mapping and aggregation operations;
3. A systematic study of the generalization capabilities of these architectures and the identification of key components for their success on this task.

7.2 Problem

We consider the setting of an embodied agent behaving in an environment. This agent interacts with the surrounding objects over time, during an episode of fixed length (T). Once this episode is over, an oracle provides exhaustive feedback in a synthetic language about everything that has happened. This language describes actions of the agent over the objects and includes spatial and temporal concepts. The spatial concepts are reference to an object through its spatial relation with others (Fig. 7.1a), and the temporal concepts are the past modality for the actions of the agent (Fig. 7.1c), past modality for spatial relations (Fig. 7.1b), and actions that unfold over time intervals. The histories of states of the agent's body and of the objects over the episode as well as the associated sentences are recorded in a buffer $\mathcal{B}$. From this setting, and echoing previous work on training agents from descriptions, we frame the Embodied Language Grounding problem as learning a parametrized truth function R_θ over couples of observations traces and sentences, tasked with predicting whether a given sentence W is true of a given episode history S or not. Formally, we aim to minimize

$$\mathbb{E}_{(S,W)\sim\mathcal{B}}\big[\mathcal{L}(R_\theta(S,W), r(S,W))\big]$$

where $\mathcal{L}$ denotes the cross-entropy loss and r denotes the ground truth boolean value for sentence W about trace S.

7.3 Temporal Playground

In the absence of any dedicated dataset providing spatio-temporal descriptions from behavioral traces of an agent, we introduce *Temporal Playground* (Fig. 7.1) an environment coupled with a templated grammar designed to study spatio-temporal language grounding. The environment is a 2D world, with procedurally-generated scene containing $N = 3$ objects sampled from 32 different object types belonging to 5 categories. Each object has a continuous 2D position, a size, a continuous color code specified by a 3D vector in RGB space, a type specified by a one-hot vector, and a boolean unit specifying whether it is grasped. The size of the object feature vector (o) is $|o| = 39$. The agent's body has its 2D position in the environment and its gripper state (grasping or non-grasping) as features (body feature vector (b) of size $|b| = 3$). In this environment, the agent can perform various actions over the length (T) of an episode. Some of the objects (the

animals) can move independently. Objects can also interact: if the agent brings food or water to an animal, it will grow in size; similarly, if water is brought to a plant, it will grow. At the end of an episode, a generative grammar generates sentences describing all the interactions that occurred. A complete specification of the environment as well as the BNF of the grammar can be found in Sup. Section C.1.2.

Synthetic language.

To enable a controlled and systematic study of how different types of spatio-temporal linguistic meanings can be learned, we argue it is necessary to first conduct a systematic study with a controlled synthetic grammar. We thus consider a synthetic language with a vocabulary of size 53 and sentences with a maximum length of 8. This synthetic language facilitates the generation of descriptions matching behavioral traces of the agent. Moreover, it allows us to express four categories of concepts associated with specific words. Thus, the generated sentences consist in four conceptual types based on the words they involve:

- **Sentences involving basic concepts.** This category of sentences talk about present-time events by referring to objects and their attributes. Sentences begin with the *'grasp'* token combined with any object. Objects can be named after their category (eg. *'animal', 'thing'*) or directly by their type (*'dog', 'door', 'algae', etc.*). Finally, the color *('red','blue','green'*) of objects can also be specified.
- **Sentences involving spatial concepts.** This category of sentences additionally involve one-to-one spatial relations and one-to-all spatial relations to refer to objects. An object can be *'left of'* another object (reference is made in relation to a single other object), or can be the *'top most'* object (reference is made in relation with all other objects). Example sentences include *'grasp thing bottom of cat'* or *'grasp thing right most'*.
- **Sentences involving temporal concepts**. This category of sentences involves talking about temporally-extended predicates and the past tense, without any spatial relations. The two temporal predicates are denoted with the words *'grow'* and *'shake'*. The truth value of these predicates can only be decided by looking at the temporal evolution of the object's size and position respectively. A predicate is transposed at the past tense if the action it describes was true at some point in the past and is no longer true in the present, this is indicated by adding the modifier *'was'* before the predicate. Example sentences include *'was grasp red chameleon'* (indicating that the agent grasped the red chameleon and then released it) and *'shake bush'*;
- **Sentences involving spatio-temporal concepts**. Finally, we consider the broad class of spatio-temporal sentences that combine spatial reference and temporal or past-tense predicates. These are sentences that involve both the spatial and temporal concepts defined above. Additionally, there is a case of where the spatial and the temporal aspects are entangled: past spatial reference. This happens when an object is referred to by its previous spatial relationship with another object. Consider the case of an animal that was at first on the bottom of a table, then moved on top, and then is grasped. In this case we could refer to this animal as

something that was previously on the bottom of the table. We use the same *'was'* modifier as for the past tense predicates; and thus we would describe the action as *'Grasp thing was bottom of table'*.

7.4 Multi-modal Transformers

In this section we describe the architectures used as well as their inputs. Let one input sample to our model be $I = (S, W)$, where $(S_{i,t})_{i,t}$ represents the objects' and body's evolution, and $(W_l)_l$ represents the linguistic observations. S has a spatial (or entity) dimension indexed by $i \in [0..N]$ and a temporal dimension indexed by $t \in [1..T]$; for any i, t, $S_{i,t}$ is a vector of observational features. Note that by convention, the trace $(S_{0,t})_t$ represents the body's features, and the traces $(S_{i,t})_{t,i>0}$ represents the other objects' features. W is a 2-dimensional tensor indexed by the sequence $l \in [1..L]$; for any l, $W_l \in \mathbb{R}^{dw}$ is a one-hot vector defining the word in the dictionary. The output to our models is a single scalar between 0 and 1 representing the probability that the sentence encoded by W is true in the observation trace S.

Transformer Architectures

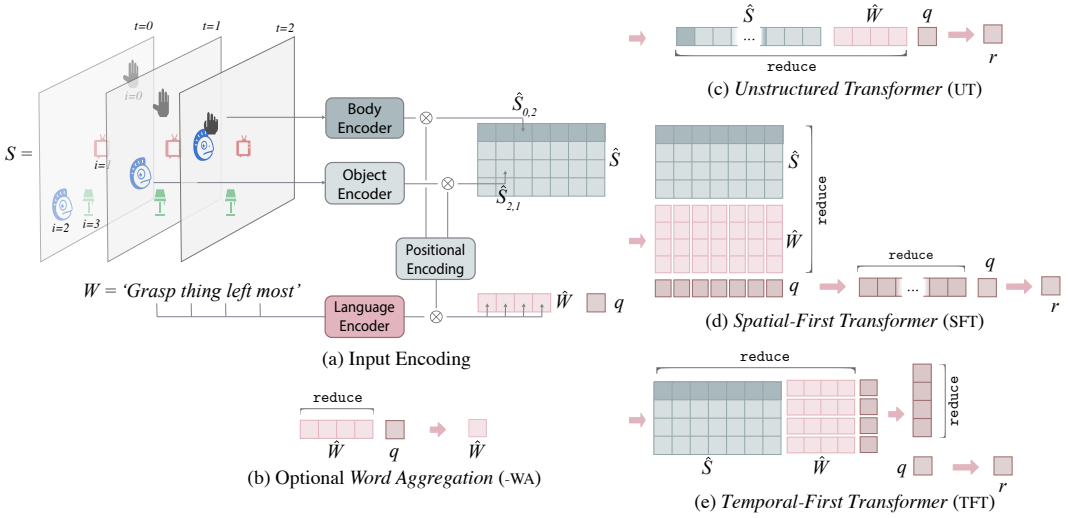


Figure 7.2: **Visual summary of the architectures used.** We show the details of UT, SFT and TFT respectively in subfigures (c), (d), (e), as well as a schematic illustration of the preprocessing phase (a) and the optional word-aggregation procedure (b).

To systematically study the influence of architectural choices on language performance and generalization in our spatio-temporal grounded language context, we define a set of mapping and aggregation operations that allows us to succinctly describe different models in a unified framework. We define:

- An aggregation operation based on a Transformer model, called `reduce`. `reduce` is a parametrized function that takes 3 inputs: a tensor, a dimension tuple D

over which to reduce and a query tensor (that has to have the size of the reduced tensor). R layers of a Transformer are applied to the input-query concatenation and are then queried at the position corresponding to the query tokens. This produces an output reduced over the dimensions D.

- A casting operation called `cast`. `cast` takes as input 2 tensors A and B and a dimension d. A is flattened, expanded so as to fit the tensor B in all dimensions except d, and concatenated along the d dimension.
- A helper expand operation called `expand` that takes as arguments a tensor and an integer n and repeats the tensor n times.

Using those operations, we define three architectures: one with no particular bias (*Unstructured Transformer*, inspired by Ding et al. (2020), or UT); one with a spatial-first structural bias – objects and words are aggregated along the spatial dimension first (*Spatial-First Transformer* or SFT); and one with a temporal-first structural bias – objects and words are aggregated along the temporal dimension first (*Temporal-First Transformer*, or TFT).

Before inputting the observations of bodies and objects S and the language W into any of the Transformer architectures, they are projected to a common dimension (see Sup. Section C.2.2 for more details). A positional encoding Vaswani et al. (2017) is then added along the time dimension for observations and along the sequence dimension for language; and finally a one-hot vector indicating whether the vector is observational or linguistic is appended at the end. This produces the modified observation-language tuple $(\hat{S}, \hat{W})$. We let:

$$\text{UT}(\hat{S}, \hat{W}) := \texttt{reduce}(\texttt{cast}(\hat{S}, \hat{W}, 0), 0, q)$$

$$\text{SFT}(\hat{S}, \hat{W}, q) := \texttt{reduce}(\texttt{reduce}(\texttt{cast}(\hat{W}, \hat{S}, 0), 0, \texttt{expand}(q, T)), 0, q)$$

$$\text{TFT}(\hat{S}, \hat{W}, q) := \texttt{reduce}(\texttt{reduce}(\texttt{cast}(\hat{W}, \hat{S}, 1), 1, \texttt{expand}(q, N+1)), 0, q)$$

where T is the number of time steps, N is the number of objects and q is a learned query token. See Fig. 7.2 for an illustration of these architectures.

Note that SFT and TFT are transpose versions of each other: SFT is performing aggregation over space first and then time, and the reverse is true for TFT. Additionally, we define a variant of each of these architectures where the words are aggregated before being related with the observations. We name these variants by appendding -WA (word-aggregation) to the name of the model (see Fig. 7.2 (b)).

$$\hat{W} \leftarrow \texttt{reduce}(\hat{W}, 0, q)$$

We examine these variants to study the effect of letting word-tokens directly interact with object-token through the self-attention layers vs simply aggregating all language tokens in a single embedding and letting this vector condition the processing of observations. The latter is commonly done in the language-conditioned RL and language grounding literature Chevalier-Boisvert et al. (2019c); Bahdanau et al. (2019b); Hui et al. (2020); Ruis et al. (2020), using the language embedding in FiLM layers Perez et al. (2017) for instance. Finding a significant effect here would encourage using architectures which allow direct interactions between the word tokens and the objects they refer to.

LSTM Baselines

We also compare some LSTM-based baselines on this task; their architecture is described in more detail in Sup. Section C.2.3.

7.4.1 Data Generation, Training and Testing Procedures

We use a bot to generate the episodes we train on. The data collected consists of 56837 trajectories of $T = 30$ time steps. Among the traces some descriptions are less frequent than others but we make sure to have at least 50 traces representing each of the 2672 descriptions we consider. We record the observed episodes and sentences in a buffer, and when training a model we sample (S, W, r) tuples with one observation coupled with either a true sentence from the buffer or another false sentence generated from the grammar. More details about the data generation can be found in Sup. Section C.2.1.

For each of the Transformer variants (6 models) and the LSTM baselines (2 models) we perform an hyper parameter search using 3 seeds in order to extract the best configuration. We extract the best condition for each model by measuring the mean F_1 on a testing set made of uniformly sampled descriptions from each of the categories define in section 7.3. We use the F_1 score because testing sets are imbalanced (the number of traces fulfilling each description is low). We then retrain best configurations over 10 seeds and report the mean and standard deviation (reported as solid black lines in Fig. 7.3 and Fig. 7.4) of the averaged F_1 score computed on each set of sentences. When statistical significance is reported in the text, it is systematically computed using a two-tail Welch's t-test with null hypothesis $\mu_1 = \mu_2$, at level $\alpha = 0.05$ **?**. Details about the training procedure and the hyper parameter search are provided in Sup. Section C.2.4.

7.5 Experiments

7.5.1 Generalization abilities of models on non-systematic split by categories of meaning

In this experiment, we perform a study of generalization to new sentences from known observations. We divide our set of test sentences in four categories based on the categories of meanings listed in Section 7.3: Basic, Spatial, Spatio-Temporal and Temporal. We remove 15% of all possible sentences in each category from the train set and evaluate the F1 score on those sentences. The results are provided in Fig. 7.3.

First, we notice that over all categories of meanings, all UT and TFT models, with or without word-aggregation, perform extremely well compared to the LSTM baselines, with all these four models achieving near-perfect test performance on the Basic sentences, with very little variability across the 10 seeds. We then notice that all SFT variants perform poorly on all test categories, in line or worse than the baselines. This is particularly visible on the spatio-temporal category, where the SFT models perform at 0.75 ± 0.020 whereas the baselines perform at 0.80 ± 0.019 . This suggests that across tasks, it is harmful to aggregate each scene plus the language information into a single vector. This

may be due to the fact that objects lose their identity in this process, since information about all the objects becomes encoded in the same vector. This may make it difficult for the network to perform computations about the truth value of predicate on a single object.

Secondly, we notice that the word-aggregation condition seems to have little effect on the performance on all three Transformer models. We only observe a significant effect for UT models on spatio-temporal concepts (p-value = 2.38e-10). This suggests that the meaning of sentences can be adequately summarised by a single vector; while maintaining separated representations for each object is important for achieving good performance it seems unnecessary to do the same for linguistic input. However we notice during our hyperparameter search that our -WA models are not very robust to hyperparameter choice, with bigger variants more sensitive to the learning rate.

Thirdly, we observe that for our best-performing models, the basic categories of meanings are the easiest, with a mean score of 1.0 ± 0.003 across all UT and TFT models, then the spatial ones at 0.96 ± 0.020, then the temporal ones at 0.96 ± 0.009, and finally the spatio-temporal ones at 0.89 ± 0.027. This effectively suggests, as we hypothesised, that sentences containing spatial relations or temporal concepts are harder to ground than those who do not.

Known sentences with novel observations

We also examine the mean performance of our models for sentences in the training set but evaluated on a set of *new observations*: we generate a new set of rollouts on the environment, and only evaluate the model on sentences seen at train time (plots are reported in Sup. Section C.3). We see the performance is slightly better in this case, especially for the LSTM baselines (0.82 ± 0.031 versus 0.79 ± 0.032), but the results are comparable in both cases, suggesting that the main difficulty for models lies in grounding spatio-temporal meanings and not in linguistic generalization for the type of generalization considered in this section.

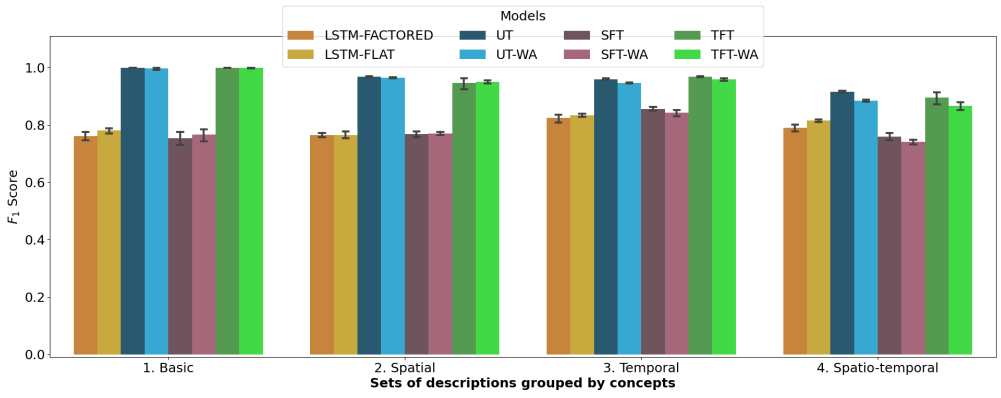


Figure 7.3: **F1 scores for all the models on randomly held-out sentences.** F_1 is measured on separated sets representing each category of concepts defined in Section 7.3.

7.5.2 Systematic generalization on withheld combinations of words

In addition to the previous generalization studies, we perform an experiment in a harder linguistic generalization setting where we systematically remove binary combinations in our train set. This is in line with previous work on systematic generalization on deep learning models Lake & Baroni (2018); Ruis et al. (2020); Hupkes et al. (2020). We create five test sets to examine the abilities of our models to generalize on binary combinations of words that have been systematically removed from the set of training sentences, but whose components have been seen before in other contexts. Our splits can be described by the set of forbidden combinations of words as:

1. **Forbidden object-attribute combinations.** remove from the train set all sentences containing *'red cat'*, *'blue door'* and *'green cactus'*. This tests the ability of models to recombine known objects with known attributes;
2. **Forbidden predicate-object combination.** remove all sentences containing *'grow'* and all objects from the *'plant'* category. This tests the model's ability to apply a known predicate to a known object in a new combination;
3. **Forbidden one-to-one relation.** remove all sentences containing *'right of'*. Since the *'right'* token is already seen as-is in the context of one-to-all relations (*'right most'*), and other one-to-one relations are observed during training, this tests the abilities of models to recombine known directions with in a known template;
4. **Forbidden past spatial relation.** remove all sentences containing the contiguous tokens *'was left of'*. This tests the abilities of models to transfer a known relation to the past modality, knowing other spatial relations in the past;
5. **Forbidden past predicate.** remove all sentences containing the contiguous tokens *'was grasp'*. This tests the ability of the model to transfer a known predicate to the past modality, knowing that it has already been trained on other past-tense predicates.

To avoid retraining all models for each split, we create one single train set with all forbidden sentences removed and we test separately on all splits. We use the same hyperparameters for all models than in the previous experiments. The results are reported in Fig. 7.4.

First we can notice that the good test scores obtained by the UT and TFT models on the previous sections are confirmed in on this experiment: they are the best performing models overall. We then notice that the first two splits, corresponding to new attribute-object and predicate-object combinations, are solved by the UT and TFT models, while the SFT models and the LSTM baselines struggle to achieve high scores. For the next 3 splits, which imply new spatial and temporal combinations, the scores overall drop significantly; we also observe much wider variability between seeds for each model, perhaps suggesting the various strategies adopted by the models to fit the train set have very different implications in terms of systematic generalization on spatial and temporal concepts. This very high variability between seeds on systematic generalization scores are reminiscent of the results obtained on the gSCAN benchmark Ruis et al. (2020).

Additionally, for split 3, which implies combining known tokens to form a new spatial relation, we observe a significant drop in generalization for the word-aggregation (WA)

conditions, consistent across models (on average across seeds, -0.14 ± 0.093, -0.15 ± 0.234 and -0.20 ± 0.061 for UT, SFT and TFT resp. with p-values $<$ 1e-04 for UT and SFT). This may be due to the fact that recombining any one-to-one relation with the known token *right* seen in the context of one-to-all relations requires a separate representation for each of the linguistic tokens. The same significant drop in performance for the WA condition can be observed for UT and TFT in split 4, which implies transferring a known spatial relation to the past.

However, very surprisingly, for split 5 – which implies transposing the known predicate *grasp* to the past tense – we observe a very strong effect in the opposite direction: the WA condition seems to help generalizing to this unknown past predicate (from close-to-zero scores for all transformer models, the *WA* adds on average 0.71 ± 0.186, 0.45 ± 0.178 and 0.52 ± 0.183 points for UT, ST and TT resp. and p-values$<$ 1e-05). This may be due to the fact that models without WA learn a direct and systematic relationship between the *grasp* token and grasped objects, as indicated in their features; this relation is not modulated by the addition of the *was* modifier as a prefix to the sentence. Models do not exhibit the same behavior on split 4, which has similar structure (transfer the relation *left of* to the past). This may be due to the lack of variability in instantaneous predicates (only the *grasp* predicate); whereas there are several spatial relations (4 one-to-one, 4 one-to-all).

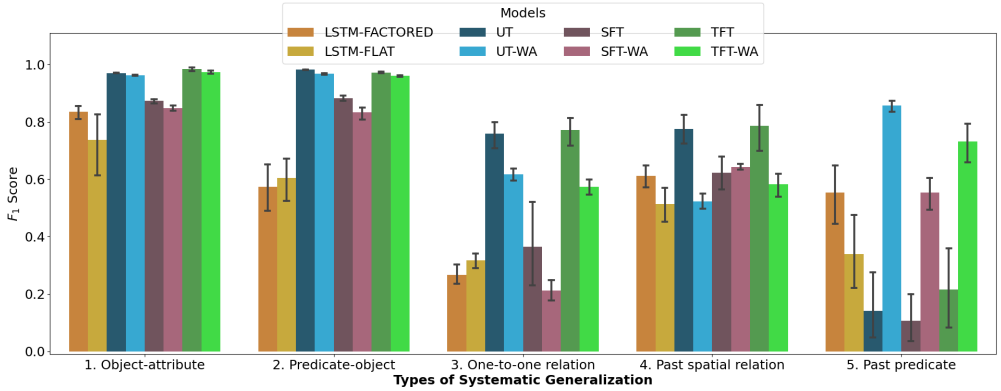


Figure 7.4: **F_1 scores of all the models on systematic generalization splits.** F_1 is measured on separated sets representing each of the forbidden combinations of word defined above.

7.6 Related Work

The idea that agents should learn to represent and ground language in their experience of the world has a long history in developmental robotics Zwaan & Madden (2005b); Steels (2006); Sugita & Tani (2005); Cangelosi et al. (2010b) and was recently extended in the context of Language Conditioned Deep Reinforcement Learning Chevalier-Boisvert et al. (2019c); **?**); Luketina et al. (2019); Bahdanau et al. (2019b). These recent approaches often consider navigation **??** or object manipulation Akakzia et al. (2021c); **?** tasks and are always using instructive language. Meanings typically refer to instantaneous

actions and rarely consider spatial reference to objects Paul et al. (2016). Although our environment includes object manipulations, we here tackle novel categories of meanings involving the grounding of spatio-temporal concepts such as the past modality or complex spatio-temporal reference to objects.

We evaluate our learning architectures on their ability to generalise to sets of descriptions that contain systematic differences with the training data so as to assess whether they correctly model grammar primitives. This procedure is similar to the *gSCAN* benchmark Ruis et al. (2020). This kind of compositional generalisation is referred as 'systematicity' by Hupkes et al. (2020). Environmental drivers that facilitate systematic generalization are also studied by Hill et al. (2020b). Although Hupkes et al. (2020) consider relational models in their work, they do not evaluate their performance on a *Language Grounding* task. Ruis et al. (2020) consider an Embodied Language Grounding setup involving one form of time-extended meanings (adverbs), but do not consider the past modality and spatio-temporal reference to objects, and do not consider learning truth functions. Also, they do not consider learning architectures that process sequences of sensorimotor observations. To our knowledge, no previous work has conducted systematic generalization studies on an Embodied Language Grounding task involving spatio-temporal language with Transformers.

The idea that relational architectures are relevant models for Language Grounding has been previously explored in the context of *Visual Reasoning*. They were indeed successfully applied for spatial reasoning in the visual question answering task *CLEVR* Santoro et al. (2017). With the recent publication of the video reasoning dataset *CLEVRER* Yi et al. (2020), those models were extended and demonstrated abilities to reason over spatio-temporal concepts, correctly answering causal, predictive and counterfactual questions Ding et al. (2020). In contrast to our study, these works around CLEVRER do not aim to analyze spatio-temporal language and therefore do not consider time-extended predicates or spatio-temporal reference to objects in their language, and do not study properties of systematic generalization over sets of new sentences.

7.7 Discussion

In this work, we have presented a first step towards learning Embodied Language Grounding of spatio-temporal concepts, framed as the problem of learning a truth function that can predict if a given sentence is true of temporally-extended observations of an agent interacting with a collection of objects. We have studied the impact of architectural choices on successful grounding of our artificial spatio-temporal language. We have modelled different possible choices for aggregation of observations and language as hierarchical Transformer architectures. We have demonstrated that in our setting, it is beneficial to process temporally-extended observations and language tokens side-by-side, as evidenced by the good score of our Unstructured Transformer variant. However, there seems to be only minimal effect on performance in aggregating temporal observations along the temporal dimension first – compared to processing all traces and the language in an unstructured manner – as long as object identity is preserved. This can inform architectural design in cases where longer episode lengths make it impossible to store all individual timesteps for each object; our experiments provide evidence that a temporal summary can be used

in these cases. Our experiments with systematic dimensions of generalization provide mixed evidence for the influence of summarizing individual words into a single vector, showing it can be detrimental to generalize to novel word combinations but also can help prevent overgeneralization of a relation between a single word and a single object without considering the surrounding linguistic context.

Limitations and further work.

There are several limitations of our setup which open important opportunities for further work. First, we have used a synthetic language that could be extended: for instance with more spatial relations and relations that are more than binary. Another axis for further research is using low-level observations. In our setting, we wanted to disentangle the effect of structural biases on learning spatio-temporal language from the problem of extracting objects from low level observations Burgess et al. (2019); Greff et al. (2020); Engelcke et al. (2020); Locatello et al. (2020); Carion et al. (2020) in a consistent manner over time (object permanence **?**Zhou et al. (2021)). Further steps in this direction are needed, and it could allow us to define richer attributes (related to material or texture) and richer temporal predicates (such as breaking, floating, etc). Finally, we use a synthetic language which is far from the richness of the natural language used by humans, but previous work has shown that natural language can be projected onto the subspace defined by synthetic language using the semantic embeddings learned by large language models Marzoev et al. (2020): this opens up be a fruitful avenue for further investigation. A further interesting future work would be to equip autotelic agents, i.e. agents that learn to represent and pursue their on goals using compositional language Colas et al. (2020b) with our proposed multi-modal transformer architecture so as to autonomously explore their environments by imagining goal narratives Bruner (1991) made of complex spatio-temporal meanings. More precisely, the agent could collect narratives by interacting with a Social Partner and align it with its recent history of observations. The agent could then use this aligned data to learn a truth function that could serve to self-train on policies matching new imagined narratives.

Chapter 8

Language as a Cognitive Tool to Imagine Goals in Curiosity Driven Exploration: IMAGINE

Contents

Abstract

Developmental machine learning studies how artificial agents can model the way children learn open-ended repertoires of skills. Such agents need to create and represent goals, select which ones to pursue and learn to achieve them. Recent approaches have considered goal spaces that were either fixed and hand-defined or learned using generative models of states. This limited agents to sample goals within the distribution of known effects. We argue that the ability to imagine out-of-distribution goals is key to enable creative discoveries and open-ended learning. Children do so by leveraging the compositionality of language as a tool to imagine descriptions of outcomes they never experienced before, targeting them as goals during play. We introduce IMAGINE, an intrinsically motivated deep reinforcement learning architecture that models this ability. Such imaginative agents, like children, benefit from the guidance of a social peer who provides language descriptions. To take advantage of goal imagination, agents must be able to leverage these descriptions

to interpret their imagined out-of-distribution goals. This generalization is made possible by modularity: a decomposition between learned goal-achievement reward function and policy relying on deep sets, gated attention and object-centered representations. We introduce the Playground environment and study how this form of goal imagination improves generalization and exploration over agents lacking this capacity. In addition, we identify the properties of goal imagination that enable these results and study the impacts of modularity and social interactions.

8.1 Motivations

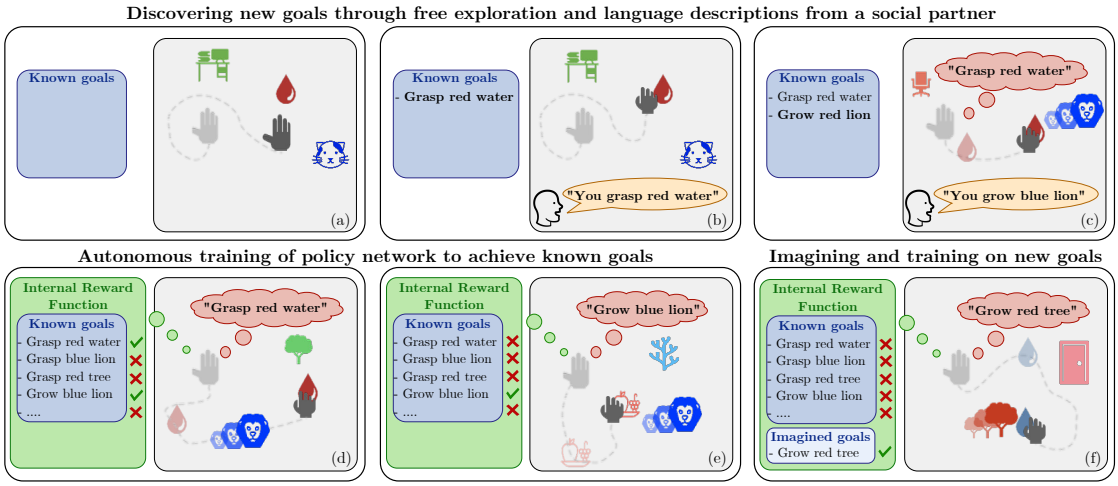


Figure 8.1: **IMAGINE overview**. In the *Playground* environment, the agent (hand) can move, grasp objects and grow some of them. Scenes are generated procedurally with objects of different types, colors and sizes. A social partner provides descriptive feedback (orange), that the agent converts into targetable goals (red bubbles).

Building autonomous machines that can discover and learn open-ended skill repertoires is a long-standing goal in Artificial Intelligence. In this quest, we can draw inspiration from children development Cangelosi & Schlesinger (2015). In particular, children exploration seems to be driven by intrinsically motivated brain processes that trigger spontaneous exploration for the mere purpose of experiencing novelty, surprise or learning progress Gopnik et al. (1999); Kaplan & Oudeyer (2007); Kidd & Hayden (2015a). During *exploratory play*, children can also invent and pursue their own problems Chu & Schulz (2020).

Algorithmic models of intrinsic motivation were successfully used in developmental robotics Oudeyer & Kaplan (2007); Baldassarre & Mirolli (2013), in reinforcement learning ?Schmidhuber (2010) and more recently in deep RL Bellemare et al. (2016); Pathak et al. (2017). Intrinsically Motivated Goal Exploration Processes (IMGEP), in particular, enable agents to sample and pursue their own goals without external rewards Baranes & Oudeyer (2013); Forestier & Oudeyer (2016); Forestier et al. (2022) and can be formulated within the deep RL framework ?Nair et al. (2018b); Colas et al. (2019a); Pong et al. (2020); Venkattaramanujam et al. (2019); ?. However, representing goal spaces and goal-achievement functions remains a major difficulty and often requires hand-crafted definitions. Past approaches proposed to learn image-based representations with generative

models such as Variational Auto-Encoders Laversanne-Finot et al. (2018); Nair et al. (2018b), but were limited to the generation of goals within the distribution of already discovered effects. Moving beyond *within-distribution* goal generation, *out-of-distribution* goal generation could power creative exploration in agents, a challenge that remains to be tackled.

In this difficult task, children leverage the properties of language to assimilate thousands of years of experience embedded in their culture, in a only a few years Tomasello (1999a); **?**. As they discover language, their goal-driven exploration changes. **?** first identified a form of egocentric speech where children narrate their ongoing activities. Later, Vygotsky (1978) realized that they were generating novel plans and goals by using the expressive generative properties of language. The harder the task, the more children used egocentric speech to plan their behavior (Vygotsky, 1978, chap. 2). Interestingly, this generative capability can push the limits of the real, as illustrated by Chomsky (1957a)'s famous example of a sentence that is syntactically correct but semantically original "*Colorless green ideas sleep furiously*". Language can thus be used to generate out-of-distributions goals by leveraging compositionality to imagine new goals from known ones.

This paper presents **I**ntrinsic **M**otivations **A**nd **G**oal **IN**vention for **E**xploration (IMAGINE): a learning architecture which leverages natural language (NL) interactions with a descriptive social partner (SP) to explore procedurally-generated scenes and interact with objects. IMAGINE discovers meaningful environment interactions through its own exploration (Figure 8.1a) and episode-level NL descriptions provided by SP (8.1b). These descriptions are turned into targetable goals by the agent (8.1c). The agent learns to represent goals by jointly training a language encoder mapping NL to goal embeddings and a goal-achievement reward function (8.1d). The latter evaluates whether the current scene satisfies any given goal. These signals (ticks in Figure 8.1d-e) are then used as training signals for policy learning. More importantly, IMAGINE can invent new goals by composing known ones (8.1f). Its internal goal-achievement function allows it to train autonomously on these imagined goals.

Related work.

The idea that language understanding is grounded in one's experience of the world and should not be secluded from the perceptual and motor systems has a long history in Cognitive Science Glenberg & Kaschak (2002a); Zwaan & Madden (2005a). This vision was transposed to intelligent systems Steels (2006); McClelland et al. (2020), applied to human-machine interaction Dominey (2005); Madden et al. (2010) and recently to deep RL via frameworks such as *BabyAI* Chevalier-Boisvert et al. (2019a).

In their review of *RL algorithms informed by NL*, Luketina et al. (2019) distinguish between *language-conditional* problems where language is required to solve the task and *language-assisted* problems where language is a supplementary help. In the first category, most works propose instruction-following agents Branavan et al. (2010); Chen & Mooney (2011); Bahdanau et al. (2019a); Co-Reyes et al. (2019); Jiang et al. (2019a); Goyal et al. (2019); Cideron et al. (2020c). Although our system is *language-conditioned*, it is not *language-instructed*: it is never given any instruction or reward but sets its own goals and

learns its own internal reward function. Bahdanau et al. (2019a) and Fu et al. (2019) also learn a reward function but require extensive expert knowledge (expert dataset and known environment dynamics respectively), whereas our agent uses experience generated by its own exploration.

Language is also particularly well suited for Hindsight Experience Replay Andrychowicz et al. (2017a): descriptions of the current state can be used to relabel trajectories, enabling agents to transfer skills across goals. While previous works used a hard-coded descriptive function Chan et al. (2019); Jiang et al. (2019a) or trained a generative model Cideron et al. (2020c) to generate goal substitutes, we leverage the learned reward function to scan goal candidates.

To our knowledge, no previous work has considered the use of compositional goal imagination to enable creative exploration of the environment. The linguistic basis of our goal imagination mechanism is grounded in construction grammar (CG). CG is a usage-based approach that characterizes language acquisition as a trajectory starting with pattern imitation and the discovery of equivalence classes for argument substitution, before evolving towards the recognition and composition of more abstract patterns Tomasello (2000); Goldberg (2003). This results in a structured inventory of constructions as form-to-meaning mappings that can be combined to create novel utterances Goldberg (2003). The discovery and substitution of equivalent words in learned schemas is observed directly in studies of child language Tomasello & Olguin (1993); Tomasello (2000). Computational implementations of this approach have demonstrated its ability to foster generalization Hinaut & Dominey (2013) and was also used for data augmentation to improve the performance of neural seq2seq models in NLP **?**.

Imagining goals by composing known ones only works in association with *systematic generalization* Bahdanau et al. (2019c); Hill et al. (2020a): generalizations of the type *grow any animal + grasp any plant → grow any plant*. These were found to emerge in instruction-following agents, including generalizations to new combinations of motor predicates, object colors and shapes Hermann et al. (2017); Hill et al. (2020a); Bahdanau et al. (2019a). Systematic generalization can occur when objects share common attributes (e.g. type, color). We directly encode that assumption into our models by representing objects as *single-slot object files* Green & Quilty-Dunn (2017): separate entities characterized by shared attributes. Because all objects have similar features, we introduce a new object-centered inductive bias: object-based modular architectures based on Deep Sets Zaheer et al. (2017).

Contributions.

This paper introduces:

1. The concept of imagining new goals using language compositionality to drive exploration.
2. IMAGINE: an intrinsically motivated agent that uses goal imagination to explore its environment, discover and master object interactions by leveraging NL descriptions from a social partner.

3. Modular policy and reward function with systematic generalization properties enabling IMAGINE to train on imagined goals. Modularity is based on Deep Sets, gated attention mechanisms and object-centered representations.

4. *Playground*: a procedurally-generated environment designed to study several types of generalizations (across predicates, attributes, object types and categories).

5. A study of IMAGINE investigating: 1) the effects of our goal imagination mechanism on generalization and exploration; 2) the identification of general properties of imagined goals required for any algorithm to have a similar impact; 3) the impact of modularity and 4) social interactions.

8.2 Playground

Open-ended learning environment.

We consider a setup where agents evolve in an environment filled with objects and have no prior on the set of possible interactions. An agent decides what and when to learn by setting its own goals, and has no access to external rewards.

However, to allow the agent to learn relevant skills, a social partner (SP) can watch the scene and plays the role of a human caregiver. Following a developmental approach Asada et al. (2009), we propose a hard-coded surrogate SP that models important aspects of the developmental processes seen in humans:

- At the beginning of each episode, the agent chooses a goal by formulating a sentence. SP then provides agents with optimal learning opportunities by organizing the scene with: 1) the required objects to reach the goal (not too difficult) 2) procedurally-generated distracting objects (not too easy and providing further discovery opportunities). This constitutes a developmental scaffolding modelling the process of Zone of Proximal Development (ZPD) introduced by Vygotsky to describe infant-parent learning dynamics Vygotsky (1978).

- At the end of each episode, SP utters a set of sentences describing achieved and meaningful outcomes (except sentences from a test set). Linguistic guidance given through descriptions are a key component of how parents "teach" language to infants, which contrasts with instruction following (providing a linguistic command and then a reward), that is rarely seen in real parent-child interactions **??**. By default, SP respects the 3 following properties: *precision*: descriptions are accurate, *exhaustiveness*: it provides all valid descriptions for each episode and *full-presence*: it is always available. Section 8.4.4 investigates relaxations of the last two assumptions.

Pre-verbal infants are known to acquire object-based representations very early Spelke et al. (1992); Johnson et al. (2003) and, later, to benefit from a simplified parent-child language during language acquisition Mintz (2003). Pursuing a developmental approach Asada et al. (2009), we assume corresponding object-based representations and a simple

grammar. As we aim to design agents that bootstrap creative exploration without prior knowledge of possible interactions or language, we do not consider the use of pre-trained language models.

Evaluation metrics.

This paper investigates how goal imagination can lead agents to efficiently and creatively explore their environment to discover interesting interactions with objects around. In this quest, SP guides agents towards a set of interesting outcomes by uttering NL descriptions. Through compositional recombinations of these sentences, goal imagination aims to drive creative exploration, to push agents to discover outcomes beyond the set of outcomes known by SP. We evaluate this desired behavior by three metrics: 1) the generalization of the policy to new states, using goals from the training set that SP knows and describes; 2) the generalization of the policy to new language goals, using goals from the testing set unknown to SP; 3) goal-oriented exploration metrics. These measures assess the quality of the agents' intrinsically motivated exploration. Measures 1) and 2) are also useful to assess the abilities of agents to learn language skills. We measure generalization for each goal as the success rate over 30 episodes and report $\overline{\texttt{SR}}$ the average over goals. We evaluate exploration with the *interesting interaction count* (I2C). I2C is computed on different sets of interesting interactions: behaviors a human could infer as goal-directed. These sets include the training, testing sets and an extra set containing interactions such as bringing water or food to inanimate objects. $\text{I2C}_{\mathcal{I}}$ measures the number of times interactions from $\mathcal{I}$ were observed over the last epoch (600 episodes), whether they were targeted or not (see Supplementary Section D.3). Thus, I2C measures the penchant of agents to explore interactions with objects around them. Unless specified otherwise, we provide means μ and standard deviations over 10 seeds and report statistical significance using a two-tail Welch's t-test with null hypothesis $\mu_1 = \mu_2$, at level $\alpha = 0.05$ (noted by star and circle markers in figures) Colas et al. (2019b).

8.3 Imagine

8.3.1 The *Playground* environment

We argue that the study of new mechanisms requires the use of controlled environments. We thus introduce *Playground*, a simple environment designed to study the impact of goal imagination on exploration and generalization by disentangling it from the problems of perception and fully-blown NL understanding. The *Playground* environment is a continuous 2D world, with procedurally-generated scenes containing $\text{N} = 3$ objects, from 32 different object types (*e.g. dog, cactus, sofa, water, etc.*), organized into 5 categories (*animals, furniture, plants, etc*), see Figure 8.1. To our knowledge, it is the first environment that introduces object categories and category-dependent combinatorial dynamics, which allows the study of new types of generalization. We release *Playground* in a separate repository.[1]

[1] https://github.com/flowersteam/playground_env

Agent perception and embodiment.

Agents have access to state vectors describing the scene: the agent's body and the objects. Each object is represented by a set of features describing its type, position, color, size and whether it is grasped. Categories are not explicitly encoded. Objects are made unique by the procedural generation of their color and size. The agent can perform bounded translations in the 2D plane, grasp and release objects with its gripper. It can make animals and plants grow by bringing them the right supply (food or water for animals, water for plants).

Grammar.

The following grammar generates the descriptions of the 256 achievable goals ($\mathcal{G}^{\text{A}}$):

1. Go: $<$*go* + ***zone***$>$ *(e.g. go bottom left)*
2. Grasp: $<$ *grasp* + *any* + ***color*** + *thing*$>$ *(e.g. grasp any blue thing)* OR $<$*grasp* + ***color*** $\cup$ {*any*} + ***object type*** $\cup$ ***object category***$>$ *(e.g. grasp red cat)*
3. Grow: $<$*grow* + *any* + ***color*** + *thing*$>$ *(e.g. grow any red thing)* OR $<$*grow* + ***color*** $\cup$ {*any*} + ***living thing*** $\cup$ {*living_thing, animal, plant*}$>$ *(e.g. grow green animal)*

Bold and { } are sets of words while *italics* are specific words. The grammar is structured around the 3 predicates *go*, *grasp* and *grow*. Objects can be referred to by a combination of their color and either their object name or category, or simply by one of these. The set of achievable goals is partitioned into *training* ($\mathcal{G}^{\text{train}}$) and *testing* ($\mathcal{G}^{\text{test}}$). $\mathcal{G}^{\text{test}}$ maximizes the compound divergence with a null atom divergence with respect to $\mathcal{G}^{\text{train}}$: testing sentences (compounds) are out of the distribution of $\mathcal{G}^{\text{train}}$ sentences, but their words (atoms) belong to the distribution of words in $\mathcal{G}^{\text{train}}$ **?**. SP only provides descriptions from $\mathcal{G}^{\text{train}}$. We limit the set of goals to better control the complexity of our environment and enable a careful study of the generalization properties. Supplementary Section D.1 provides more details about the environment, the grammar and SP as well as the pseudo-code of our learning architecture.

8.3.2 The IMAGINE Architecture

IMAGINE agents build a repertoire of goals and train two internal models: 1) a goal-achievement reward function $\mathcal{R}$ to predict whether a given description matches a behavioral trajectory; 2) a policy π to achieve behavioral trajectories matching descriptions. The architecture is presented in Figure 8.2 and follows this logic:

1. The *Goal Generator* samples a target goal g_{target} from known and imagined goals ($\mathcal{G}_{\text{known}} \cup \mathcal{G}_{\text{im}}$).
2. The agent (*RL Agent*) interacts with the environment using its policy π conditioned on g_{target}.

3. State-action trajectories are stored in a replay buffer $mem(\pi)$.
4. SP's descriptions of the last state are considered as potential goals $\mathcal{G}_{\text{SP}}(\mathbf{s}_T) = \mathcal{D}_{\text{SP}}(\mathbf{s}_T)$.
5. $mem(\mathcal{R})$ stores positive pairs $(\mathbf{s}_T,\ \mathcal{G}_{\text{SP}}(\mathbf{s}_T))$ and infers negative pairs $(\mathbf{s}_T,\ \mathcal{G}_{\text{known}} \setminus \mathcal{G}_{\text{SP}}(\mathbf{s}_T))$.
6. The agent then updates:
   - *Goal Gen.*: $\mathcal{G}_{\text{known}} \leftarrow \mathcal{G}_{\text{known}} \cup \mathcal{G}_{\text{SP}}(\mathbf{s}_T)$ and $\mathcal{G}_{im} \leftarrow \text{Imagination}(\mathcal{G}_{\text{known}})$.
   - *Language Encoder* (L_e) and *Reward Function* ($\mathcal{R}$) are updated using data from $mem(\mathcal{R})$.
   - *RL agent*: We sample a batch of state-action transitions $(\mathbf{s},\ \mathbf{a},\ \mathbf{s}')$ from $mem(\pi)$. Then, we use *Hindsight Replay* and $\mathcal{R}$ to bias the selection of substitute goals to train on (g_s) and compute the associated rewards $(\mathbf{s},\ \mathbf{a},\ \mathbf{s}',\ g_s,\ r)$. Substituted goals g_s can be known or imagined goals. Finally, the policy and critic are trained via RL.

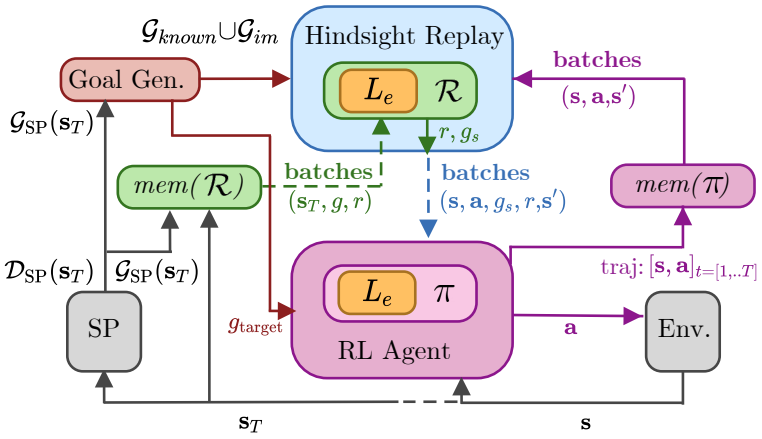


Figure 8.2: **IMAGINE architecture.** Colored boxes show the different modules of IMAGINE. Lines represent update signals (dashed) and function outputs (plain). The language encoder L_e is shared.

Goal generator.

It is a generative model of NL goals. It generates target goals g_{target} for data collection and substitutes goals g_{s} for hindsight replay. When goal imagination is disabled, the goal generator samples uniformly from the set of known goals $\mathcal{G}_{\text{known}}$, sampling random vectors if empty. When enabled, it samples with equal probability from $\mathcal{G}_{\text{known}}$ and $\mathcal{G}_{\text{im}}$ (set of imagined goals). $\mathcal{G}_{\text{im}}$ is generated using a mechanism grounded in construction grammar that leverages the compositionality of language to imagine new goals from $\mathcal{G}_{\text{known}}$. The heuristic consists in computing sets of *equivalent words*: words that appear in two sentences that only differ by one word. For example, from *grasp red lion* and *grow red lion*, *grasp* and *grow* can be considered *equivalent* and from *grasp green tree* one can imagine a new goal *grow green tree* (see Figure 8.1f). Imagined goals do not include

known goals. Among them, some are meaningless, some are syntactically correct but infeasible (e.g. *grow red lamp*) and some belong to $\mathcal{G}^{\text{test}}$, or even to $\mathcal{G}^{\text{train}}$ before they are encountered by the agent and described by SP. The pseudo-code and all imaginable goals are provided in Supplementary Section D.4.

Language encoder.

The language encoder (L_e) embeds NL goals ($L_e : \mathcal{G}^{\text{NL}} \rightarrow \mathbb{R}^{100}$) using an LSTM Hochreiter & Schmidhuber (1997) trained jointly with the reward function. L_e acts as a goal translator, turning the goal-achievement reward function, policy and critic into language-conditioned functions.

Object-centered modular architectures.

The goal-achievement reward function, policy and critic leverage novel *modular-attention* (MA) architectures based on Deep Sets Zaheer et al. (2017), gated attention mechanisms Chaplot et al. (2018) and object-centered representations. The idea is to ensure efficient skill transfer between objects, no matter their position in the state vector. This is done through the combined use of a shared neural network that encodes object-specific features and a permutation-invariant function to aggregate the resulting latent encodings. The shared network independently encodes, for each object, an affordance between this object (object observations), the agent (body observations) and its current goal. The goal embedding, generated by L_e, is first cast into an attention vector in $[0,\ 1]$, then fused with the concatenation of object and body features via an Hadamard product (gated-attention Chaplot et al. (2018)). The resulting object-specific encodings are aggregated by a permutation-invariant function and mapped to the desired output via a final network (e.g. into actions or action-values). Supplementary Section D.5 provides visual representations.

Reward function.

Learning a goal-achievement reward function ($\mathcal{R}$) is framed as binary classification: $\mathcal{R}(\mathbf{s}, \mathbf{g}) : \mathcal{S} \times \mathbb{R}^{100} \rightarrow \{0, 1\}$. We use the MA architecture with attention vectors $\boldsymbol{\alpha}^g$, a shared network $\text{NN}^{\mathcal{R}}$ with output size 1 and a logical OR aggregation. $\text{NN}^{\mathcal{R}}$ computes object-dependent rewards r_i in $[0, 1]$ from the object-specific inputs and the goal embedding. The final binary reward is computed by NN^{OR} which outputs 1 whenever $\exists j : r_j > 0.5$. We pre-trained a neural-network-based OR function to enable end-to-end training with back-propagation. The overall function is:

$$\mathcal{R}(\mathbf{s}, g) = \text{NN}^{\text{OR}}([\text{NN}^{\mathcal{R}}(\mathbf{s}_{obj(i)} \odot \boldsymbol{\alpha}^g)]_{i \in [1..N]})$$

Data. Interacting with the environment and SP, the agent builds a set of entries $[\mathbf{s}_T,\ g,\ r]$ with $g \in \mathcal{G}_{\text{known}}$ where $r \in \{0,\ 1\}$ rewards the achievement of g in state $\mathbf{s}_T$: $r = 1$ if $g \in \mathcal{G}_{\text{SP}}(\mathbf{s}_T)$ and 0 otherwise. L_e and $\mathcal{R}$ are periodically updated jointly by back-propagation on this dataset.

Multi-goal RL agent.

Our agent is controlled by a goal-conditioned policy π Schaul et al. (2015a) based on the MA architecture (see Supplementary Figure D.9b). It uses an attention vector $\boldsymbol{\beta}^g$, a shared network NN^π, a sum aggregation and a mapper NN^a that outputs the actions. Similarly, the critic produces action-values via $\boldsymbol{\gamma}^g$, NN^Q and $\text{NN}^\text{a-v}$ respectively:

$$\pi(\mathbf{s}, g) \;=\; \text{NN}^\text{a}\Big(\sum_{i\in[1..N]} \text{NN}^\pi(\mathbf{s}_{obj(i)} \odot \boldsymbol{\beta}^g)\Big) \qquad Q(\mathbf{s}, \mathbf{a}, g) \;=\; \text{NN}^\text{a-v}\Big(\sum_{i\in[1..N]} \text{NN}^Q([\mathbf{s}_{obj(i)},\ \mathbf{a}] \odot \boldsymbol{\gamma}^g)\Big).$$

Both are trained using DDPG Lillicrap et al. (2016), although any other off-policy algorithm can be used. As detailed in Supplementary Section D.6, our agent uses a form of Hindsight Experience Replay Andrychowicz et al. (2017a).

8.4 Experiments

This section first showcases the impact of goal imagination on exploration and generalization (Section 8.4.1). For a more complete picture, we analyze other goal imagination mechanisms and investigate the properties enabling these effects (Section 8.4.2). Finally, we show that our modular architectures are crucial to a successful goal imagination (Section 8.4.3) and discuss more realistic interactions with SP (Section 8.4.4). IMAGINE agents achieve near perfect generalizations to new states (training set of goals): $\overline{\texttt{SR}} \;=\; 0.95 \pm 0.05$. We thus focus on language generalization and exploration. Supplementary Sections D.2 to D.7 provide additional results and insights organized by theme (Generalization, Exploration, Goal Imagination, Architectures, Reward Function and Visualizations).

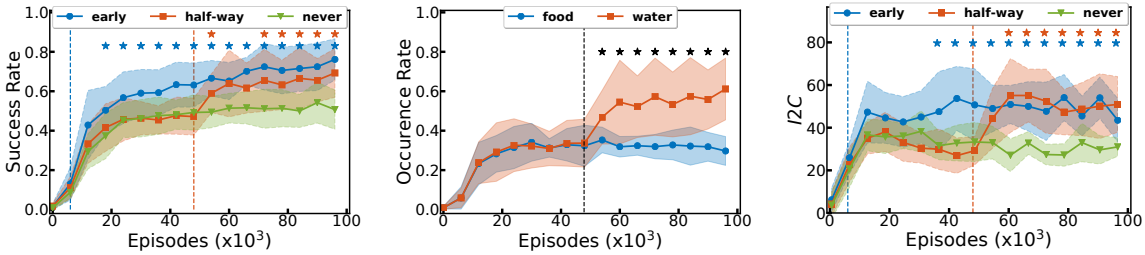
8.4.1 How does Goal Imagination Impact Generalization and Exploration?

Global generalization performance.

Figure **??** shows $\overline{\texttt{SR}}$ on the set of testing goals, when the agent starts imagining new goals early (after $6 \cdot 10^3$ episodes), half-way (after $48 \cdot 10^3$ episodes) or when not allowed to do so. Imagining goals leads to significant improvements in generalization.

A particular generalization: growing plants.

Agents learn to grow animals from SP's descriptions, but are never told they could grow plants. When evaluated offline on the *growing-plants* goals before goal imagination, agents' policies perform a sensible zero-shot generalization and bring them water or food with equal probability, as they would do for animals (Figure 8.3, left). As they start to imagine and target these goals, their behavior adapts (Figure 8.3, right). If the



(a) (b) (c)

Figure 8.3: **Goal imagination drives exploration and generalization.** Vertical dashed lines mark the onset of goal imagination. (a) $\overline{\text{SR}}$ on testing set. (b) Behavioral adaptation, empirical probabilities that the agent brings supplies to a plant when trying to grow it. (c) I2C computed on the testing set. Stars indicate significance (a and c are tested against *never*).

reward function shows good zero-shot abilities, it only provides positive rewards when the agent brings water. The policy slowly adapts to this internal reward signal and pushes agents to bring more water. We call this phenomenon *behavioral adaptation*. Supplementary Section D.2 details the generalization abilities of IMAGINE for 5 different types of generalizations involving predicates, attributes and categories.

Exploration.

Figure **??** presents the I2C metric computed on the set of interactions related to $\mathcal{G}^{\text{test}}$ and demonstrates the exploration boost triggered by goal imagination. Supplementary Section D.3 presents other I2C metrics computed on additional interactions sets.

8.4.2 What If We Used Other Goal Imagination Mechanisms?

Properties of imagined goals.

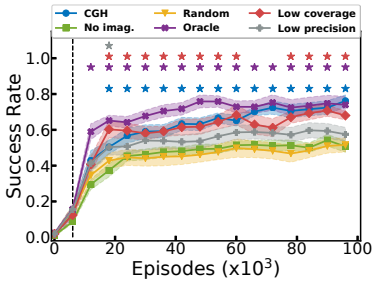
We propose to characterize goal imagination mechanisms by two properties: 1) *Coverage*: the fraction of $\mathcal{G}^{\text{test}}$ found in $\mathcal{G}_{\text{im}}$ and 2) *Precision*: the fraction of the imagined goals that are achievable. We compare our goal imagination mechanism based on the construction grammar heuristic (CGH) to variants characterized by 1) lower coverage; 2) lower precision; 3) perfect coverage and precision (oracle); 4) random goal imagination baseline (random sequences of words from $\mathcal{G}^{\text{train}}$ leading to near null coverage and precision). These measures are computed at the end of experiments, when all goals from $\mathcal{G}^{\text{train}}$ have been discovered (Figure 8.4a).

Figure 8.4b shows that CGH achieves a generalization performance on par with the oracle. Reducing the coverage of the goal imagination mechanism still brings significant improvements in generalization. Supplementary Section D.4 shows, for the *Low Coverage* condition, that the generalization performance on the testing goals that were imagined is not statistically different from the performance on similar testing goals that could have been imagined but were not. This implies that the generalization for imagined goals also benefits similar non-imagined goals from $\mathcal{G}^{\text{test}}$. Finally, reducing the precision of imagined

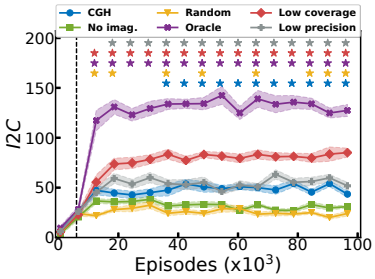
goals (gray curve) seems to impede generalization (no significant difference with the *no imagination* baseline). Figure 8.4c shows that all goal imagination heuristics enable a significant exploration boost. The random goal baseline acts as a control condition. It demonstrates that the generalization boost is not due to a mere effect of network regularization introduced by adding random goals (no significant effect w.r.t. the *no imagination* baseline). In the same spirit, we also ran a control using random goal embeddings, which did not produce any significant effects.

(a)

	Cov.	Pre.
CGH	0.87	0.45
Oracle	1	1
Low Cov.	0.44	0.45
Low Pre.	0.87	0.30
Random G.	≈0	≈0



(b)



(c)

Figure 8.4: **Goal imagination properties.** (a) Coverage and precision of different goal imagination heuristics. (b) $\overline{\text{SR}}$ on testing set. (c) I2C on $\mathcal{G}^{\text{test}}$. We report *sem* (standard error of the mean) instead of *std* to improve readability. Stars indicate significant differences w.r.t the *no imagination* condition.

8.4.3 How Does Modularity Interact with Goal Imagination?

Table 8.1: Policy architectures performance. $\overline{\text{SR}}_{\text{test}}$ at convergence.

	MA *	FA
Im.	0.76 ± 0.1	0.15 ± 0.05
No Im.	0.51 ± 0.1	0.17 ± 0.04
p-val	4.8e-5	0.66

We compared MA to flat architectures (FA) that consider the whole scene at once. As the use of FA for the reward function showed poor performance on $\mathcal{G}^{\text{train}}$, Table 8.1 only compares the use of MA and FA for the policy. MA shows stronger generalization and is the only architecture allowing an additional boost with goal imagination. Only MA policy architectures can leverage the novel reward signals coming from imagined goals and turn them into *behavioral adaptation*. Supplementary Section D.5 provides additional details.

8.4.4 Can We Use More Realistic Feedbacks?

We study the relaxation of the *full-presence* and *exhaustiveness* assumptions of SP. We first relax *full-presence* while keeping *exhaustiveness* (blue, yellow and purple curves).

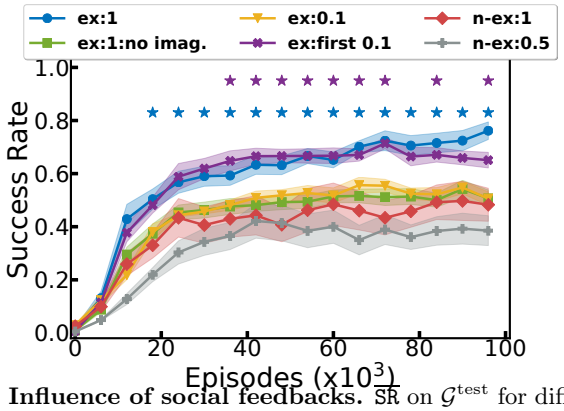


Figure 8.5: **Influence of social feedbacks.** SR on $\mathcal{G}^{\text{test}}$ for different social strategies. Stars indicate significant differences w.r.t. *ex:1 no imag.*. sem plotted, 5 seeds.

When SP has a 10% chance of being present (yellow), imaginative agents show generalization performance on par with the unimaginative agents trained in a full-presence setting (green), see Figure 8.5). However, when the same amount of feedback is concentrated in the first 10% episodes (purple), goal imagination enables significant improvements in generalization (w.r.t. green). This is reminiscent of children who require less and less attention as they grow into adulthood and is consistent with Chan et al. (2019). Relaxing *exhaustiveness*, SP only provides one positive and one negative description every episode (red) or in 50% of the episodes (gray). Then, generalization performance matches the one of unimaginative agents in the exhaustive setting (green).

8.5 Discussion and Conclusion

IMAGINE is a learning architecture that enables autonomous learning by leveraging NL interactions with a social partner. As other algorithms from the IMGEP family, IMAGINE sets its own goals and builds behavioral repertoires without external rewards. As such, it is distinct from traditional instruction-following RL agents. This is done through the joint training of a language encoder for goal representation and a goal-achievement reward function to generate internal rewards. Our proposed modular architectures with gated-attention enable efficient out-of-distribution generalization of the reward function and policy. The ability to imagine new goals by composing known ones leads to further improvements over initial generalization abilities and fosters exploration beyond the set of interactions relevant to SP. Our agent even tries to grow pieces of furniture with supplies, a behavior that can echo the way a child may try to feed his doll.

IMAGINE does not need externally-provided rewards but learns which behaviors are *interesting* from language-based interactions with SP. In contrast with hand-crafted reward functions, NL descriptions provide an easy way to guide machines towards relevant interactions. *A posteriori* counterfactual feedback is easier to communicate for humans, especially when possible effects are unknown and, thus, the set of possible instructions is undefined. Hindsight learning also greatly benefits from such counterfactual feedback and improves sample efficiency. Attention mechanisms further extend the interpretability of the agent's learning by mapping language to attentional scaling factors (see Supplementary Figure D.12). In addition, Section 8.4.4 shows that agents can learn to achieve goals

from a relatively small number of descriptions, paving the way towards human-provided descriptions.

Playground is a tool that we hope will enable the community to further study under-explored descriptive setups with rich combinatorial dynamics, as well as goal imagination. It is designed for the study of goal imagination and combinatorial generalization. Compared to existing environments (Hermann et al., 2017; Chevalier-Boisvert et al., 2019a; Chan et al., 2019), we allow the use of descriptive feedback, introduce the notion of object categories and category-dependent object interactions (*Grow* refer to different modalities for *plants* or *animals*). Playground can easily be extended by adding objects, attributes, category- or object-type-dependent dynamics.

IMAGINE could be combined with unsupervised multi-object representation learning algorithms Burgess et al. (2019); Greff et al. (2019) to work directly from pixels, practically enforcing object-centered representations. The resulting algorithm would still be different from goal-as-state approaches Nair et al. (2018b); Pong et al. (2020); **?**. Supplementary Section D.8 discusses the relevance of comparing IMAGINE to these works. Some tasks involve instruction-based navigation in visual environments that do not explictly represent objects (**??**). Here, also, imagining new instructions from known ones could improve exploration and generalization. Finally, we believe IMAGINE could provide interesting extensions in hierarchical settings, like in Jiang et al. (2019a), with novel goal imagination boosting low-level exploration.

Future work.

A more complex language could be introduced, for example, by considering object relationships (e.g. *Grasp any X left of Y*), see **?** for a preliminary experiment in this direction. While the use of pre-trained language models Radford et al. (2019) does not follow our developmental approach, it would be interesting to study how they would interact with goal imagination. Because CGH performs well in our setup with a medium precision (0.45) and because similar mechanisms were successfully used for data augmentation in complex NLP tasks **?**, we believe our goal imagination heuristic could scale to more realistic language.

We could reduce the burden on SP by considering unreliable feedbacks (lower precision), or by conditioning goal generation on the initial scene (e.g. using mechanisms from Cideron et al. (2020c)). One could also add new interaction modalities by letting SP make demonstrations, propose goals or guide the agent's attention. Our modular architectures, because they are set functions, could also directly be used to consider variable numbers of objects. Finally, we could use off-policy learning Fujimoto et al. (2019) to reinterpret past experience in the light of new imagined goals without any additional environment interactions.

Links.

Demonstration videos are available at `https://sites.google.com/view/imagine-drl`. The source code of playground environment can be found at `https://github.`

com/flowersteam/playground_env and the source code of the IMAGINE architecture https://github.com/flowersteam/Imagine.

Part III

Discussion

Chapter 9

Challenges

Contents

9.1 Autotelic Agents Challenges

Todo:*this needs to be pruned and reformulated* This section discusses open challenges in the quest for autotelic agents tackling the intrinsically motivated skills acquisition problem.

9.1.1 Challenge #1: Targeting a Greater Diversity of Goals

Section 6.2.3 introduces a typology of goal representations found in the literature. The diversity of goal representations seems however limited, compared to the diversity of goals human target Ram et al. (1995).

Time-extended goals.

All RL approaches reviewed in this paper consider *time-specific* goals, that is, goals whose completion can be assessed from any state s. This is due to the Markov property requirement, where the next state and reward need to be a function of the previous state only. *Time-extended* goals — i.e. goals whose completion can be judged by observing a sequence of states (e.g. *jump twice*) — can however be considered by adding time-extended features to the state **?**e.g. the difference between the current state and the initial state>imagine. To avoid such *ad-hoc* state representations, one could imagine

using reward function architectures that incorporate forms of memory such as Recurrent Neural Network (RNN) architectures Elman (1993) or Transformers Vaswani et al. (2017). Although recurrent policies are often used in the literature Chevalier-Boisvert et al. (2019a); Hill et al. (2020a); Loynd et al. (2020); Goyal et al. (2021), recurrent reward functions have not been much investigated. Some work Sutton & Tanner (2004); Schlegel et al. (2021) investigate the benefit of computing relations between value functions when learning predictive representations. Sutton & Tanner (2004) propose to represent the interrelation of predictions in a *TD-network* where nodes are predictions computed from states. The network allows to perform predictions that have complex temporal semantics. Schlegel et al. (2021) train a RNN architecture where hidden-states are multi-step predictions. Finally, recent work by Karch et al. (2021) show that agents can derive rewards from linguistic descriptions of time-extended behaviors. Time-extended goals include interactions that span over multiple time steps (e.g. *shake the blue ball*) and spatio-temporal references to objects (e.g. *get the red ball that was on the left of the sofa yesterday*).

Learning goals.

Goal-driven learning is the idea that humans use *learning goals*, goals about their own learning abilities as a way to simplify the realization of *task goals* Ram et al. (1995). Here, we refer to *task goals* as goals that express constraints on the physical state of the agent and/or environment. On the other hand, *learning goals* refer to goals that express constraints on the knowledge of the agent. Although most RL approaches target task goals, one could envision the use of *learning goals* for RL agents.

In a way, learning-progress-based learning is a form of learning goal: as the agent favors regions of the goal space to sample its task goals, it formulates the goal of learning about this specific goal region Baranes & Oudeyer (2013); Fournier et al. (2018, 2021); Colas et al. (2019a); Blaes et al. (2019); Akakzia et al. (2021a).

Embodied Question Answering problems can also be seen as using learning goals. The agent is asked a question (i.e. a learning goal) and needs to explore the environment to answer it (acquire new knowledge) Das et al. (2018b); Yuan et al. (2019b).

In the future, one could envision agents that set their own learning targets as subgoals towards the resolution of harder task or learning goals, e.g. *I'm going to learn about knitting so I can knit a pullover to my friend for his birthday.*

Goals as optimization under selected constraints.

We discussed the representations of goals as a balance between multiple objectives. An extension of this idea is to integrate the selection of constraints on states or trajectories. One might want to maximize a given metric (e.g. walking speed), while setting various constraints (e.g. maintaining the power consumption below a given threshold or controlling only half of the motors). The agent could explore in the space of constraints, setting constraints to itself, building a curriculum on these, etc. This is partially investigated in Colas et al. (2021a), where the agent samples constraint-based goals in the optimization of control strategies to mitigate the economic and health costs in simulated

epidemics. This approach, however, only considers constraints on minimal values for the objectives and requires the training of an additional Q-function per constraint.

Meta-diversity of goals.

Finally, autotelic agents should learn to target all these goals within the same run; to transfer their skills and knowledge between different types of goals. For instance, targeting visual goals could help the agent explore the environment and solve learning goals or linguistic goals. As the density of possible goals increases, agents can organize more interesting curricula. They can select goals in easier representation spaces first (e.g. sensorimotor spaces), then move on to target more difficult goals (e.g. in the visual space), before they can target the more abstract goals (e.g. learning goals, abstract linguistic goals).

This can take the form of goal spaces organized hierarchically at different levels of abstractions. The exploration of such complex goal spaces has been called *meta-diversity* Etcheverry et al. (2020). In the outer-loop of the meta-diversity search, one aims at learning a diverse set of outcome/goal representations. In the inner-loop, the exploration mechanism aims at generating a diversity of behaviors in each existing goal space. How to efficiently transfer knowledge and skills between these multi-modal goal spaces and how to efficiently organize goal selection in large multi-modal goal spaces remains an open question.

9.1.2 Challenge #2: Learning to Represent Diverse Goals

This survey mentioned only a handful of complete autotelic architectures. Indeed, most of the surveyed approach assume pre-existing goal embeddings or reward functions. Among the approaches that learn goal representations autonomously, we find that the learned representations are often restricted to very specific domains. Visual goal-conditioned approaches for example, learn reward functions and goal embeddings but restrict them to the visual space Nair et al. (2018b, 2020); Warde-Farley et al. (2019); Venkattaramanujam et al. (2019); Pong et al. (2020); Hartikainen et al. (2020). Empowerment methods, on the other hand, develop skills that maximally cover the state space, often restricted to a few of its dimensions Achiam et al. (2018); Eysenbach et al. (2019); Campos et al. (2020); Sharma et al. (2020).

These methods are limited to learn goal representations within a bounded, pre-defined space: the visual space, or the (sub-) state space. How to autonomously learn to represent the wild diversity of goals surveyed in Section 6.2.3 and discussed in Challenge #1 remains an open question.

9.1.3 Challenge #3: Imagining Creative Goals

Goal sampling methods surveyed in Section 6.2.5 are all bound to sample goals *within the distribution of known effects*. Indeed, the support of the goals distribution is either pre-defined **?**e.g. >schaul2015universal,andrychowicz2017hindsight,curious,li2019towards or learned using a generative model Florensa et al. (2018); Nair et al. (2018b, 2020); Pong

et al. (2020) trained on previously experienced outcomes. On the other hand, humans can imagine creative goals beyond their past experience which, arguably, powers their exploration of the world.

In this survey, one approach opened a path in this direction. The IMAGINE algorithm uses linguistic goal representation learned via social supervision and leverages the compositionality of language to imagine creative goals beyond its past experience Colas et al. (2020b). This is implemented by a simple mechanism detecting templates in known goals and recombining them to form new ones. This is in line with a recent line of work in developmental psychology arguing that human play might be about practicing to generate plans to solve imaginary problems Chu & Schulz (2020).

Another way to achieve similar outcomes is to compose known goals with Boolean algebras, where new goals can be formed by composing existing atomic goals with negation, conjunction and disjunctions. The logical combinations of atomic goals was investigated in Tasse et al. (2020); Chitnis et al. (2021), and Colas et al. (2020a); Akakzia et al. (2021a). The first approach represents the space of goals as a Boolean algebra, which allows immediate generalization to compositions of goals (AND, OR, NOT). The second approach considers using general symbolic and logic languages to express goals, but uses symbolic planning techniques that are not yet fully integrated in the goal-conditioned deep RL framework. The third and fourth train a generative model of goals conditioned on language inputs. Because it generates discrete goals, it can compose language instructions by composing the finite sets of discrete goals associated to each instruction (AND is the intersection, OR the union etc). However, these works fall short of exploring the richness of goal compositionality and its various potential forms. Tasse et al. (2020) seem to be limited to specific goals as target features, while Akakzia et al. (2021a) requires discrete goals. Finally, Barreto et al. (2019) proposes to target new goals that are represented by linear combination of pseudo-rewards called *cumulants*. They use the option framework and show that an agent that masters a set of options associated with cumulants can generalize to any new behavior induced by a linear combination of those known cumulants.

9.1.4 Challenge #4: Composing Skills for Better Generalization

Although this survey focuses on goal-related mechanisms, autotelic agents also need to learn to achieve their goals. Progress in this direction directly relies on progress in standard RL and goal-conditioned RL. In particular, autotelic agents would considerably benefit from better generalization and skill composition. Indeed, as the set of goals agents can target grows, it becomes more and more crucial that agents can efficiently transfer knowledge between skills, infer new skills from the ones they already master and compose skills to form more complex ones. Although hierarchical RL approach learn to compose skills sequentially, concurrent skill composition remains under-explored.

9.1.5 Challenge #6: Leveraging Socio-Cultural Environments

Decades of research in psychology, philosophy, linguistics and robotics have demonstrated the crucial importance of rich socio-cultural environments in human development Vygotsky (1934); Whorf (1956); Wood et al. (1976); Rumelhart et al. (1986); Berk (1994);

Clark (1998b); Tomasello (1999b, 2009); Zlatev (2001); Carruthers (2002); Dautenhahn et al. (2002); Lindblom & Ziemke (2003); Mirolli & Parisi (2011); Lupyan (2012). However, modern AI may have lost track of these insights. Deep reinforcement learning rarely considers social interactions and, when it does, models them as direct teaching; depriving agents of all autonomy. A recent discussion of this problem and an argument for the need of agents that are both autonomous and teachable can be found in a concurrent work Sigaud et al. (2021a). As we embed autotelic agents in richer socio-cultural worlds and let them interact with humans, they might start to learn goal representations that are meaningful for us, in our society.

9.2 Vygotksian Agents challenges

We identify three main challenges for future research.

Challenge #1: Immersing autotelic agents in rich socio-cultural worlds. To benefit from language, Vygotskian autotelic agents must be immersed into rich socio-cultural worlds close to ours. This will require progress along two dimensions: 1) increasing the richness of their world and 2) augmenting their interactivity and teachability.

What do we mean by *rich* worlds? One aspect is the multimodality of perceptions. Beyond its linguistic dimension, culture is indeed multimodal. Socio-cultural interactions are not always linguistic but often non-verbal as they may involve motor, perceptual or emotional dimensions. The second aspect is socio-cultural situatedness: autotelic agents must interact with other agents and with humans. Scaling the richness of these worlds may thus require the involvement of the video game industry, and specialists in complex, realistic multimodal worlds. Human-in-the-loop research will also be required to let humans enter these rich virtual worlds, for instance via virtual reality technology.

Vygotskian autotelic agents will need to be more interactive and teachable. In a recent paper, Sigaud and colleagues discuss this challenge through a detailed analysis of children's learning abilities and teacher-child interactions.Sigaud et al. (2021b) They present a checklist of properties that future Vygotskian autotelic agents must demonstrate to be considered *teachable*. To interact with humans, Vygotskian autotelic agents will also need to target goals in multiple modalities (e.g. linguistic, perceptual, emotional) with various levels of abstraction.Colas et al. (2022) Modular autotelic architectures may be used to that end. By handling multiple goal spaces in parallel, they can leverage cross-domain hindsight learning: using experience collected while aiming at a goal to learn about other goals in other domains.Colas et al. (2019a)

Challenge #2: Enabling artificial mental life with systematic internalized language production Only a few approaches internalize language production within agents. So far limited to a few use cases, language production should concern every possible linguistic feedback agents could receive: instructions, corrections, advice, explanations, or cultural artefacts. This internal language production is akin to an artificial *inner speech*, the embryo of *artificial mental life*. Looping back to the constitutive thesis of Carruthers presented in Section 6.3.1, inner speech acts as a common currency for inner modules to exchange information (see a recent implementation of this ideaZeng et al. (2022)). Combined with world models, inner speech could trigger the simulation

of perceptual experience (images, sounds), sensorimotor trajectories, the imagination of possible futures or past memories. Observing these hallucinations, agents could produce new behaviors and new inner speech. This inner loop acts as a mental life that could help agents reason; trigger memories or mnemotechnic representations acting as cognitive aids. As noted by Dove, this account is fully compatible with the embodied hypothesis in cognitive science.Dove (2018) Following this hypothesis, thinking and modeling sensorimotor experience are one and the same. Here, language brings another set of inputs and outputs for these simulation models and the simulation of abstract content (words, analogical structures, etc) might offer us the capacity to reason abstractly.

Challenge #3: Building editable and shareable cultural models with aligned LLMs. Large language models encode a lot of information about the human cultures that generated the texts they were trained on LLMs,West et al. (2022); Schramowski et al. (2022). They can be viewed as (partial) cultural models: by tapping into them, agents could learn about human cultures. They could learn about foundational human concepts, causality, folk psychology, politeness, ethics and all these physical or cultural information that are the subject of everyday stories: fiction, news, or even simple narratives parents use to explain everyday things to children.

Such proxies to human cultures provide both opportunities and challenges. Using existing LLMs as is, we could for example prompt them to generate new goals for exploration or even full curricula based on descriptions of the agent's current abilities and environmental descriptions. We could use LLMs to predict the outcome of the agent's actions given the context and use this to plan in abstract search spaces. We could let agents ask LLMs for guidelines only when they cannot solve the problem themselves (active learning), and more generally to augment the world state with commonsense knowledge.

But letting autotelic agents rely on LLMs might also bring some downsides. Cultural information, because it biases the search space, may limit exploration and lead to the premature abandonment of promising avenuesBonawitz et al. (2011) (e.g. in astronomy, the cultural support for the geocentric model significantly delayed the acceptance of the heliocentric model). LLMs are also known to convey false information and harmful biases, either because they inadequately learned to encode a culture or because they were trained on cultural artefacts which contained such biases.Shah et al. (2020); Liang et al. (2021); Weidinger et al. (2021); Bender et al. (2021) Autotelic agents relying on these models could demonstrate harmful behaviors and contribute to reinforcing stereotypes and inequalities. The use of LLMs will thus require advances in bias mitigation strategiesLiang et al. (2021); Bender et al. (2021) and improved alignment methods to make LLMs more reliable, trustworthy and moral. Ideally, we want them to model the natural culture agents will be embedded in with high fidelity and align well with its objectives.

Humans are also biased by the cultural environment they are in. During their education, children are taught to think for themselves and to think critically. Autotelic agents should be taught in the same way. Because they are autonomous embodied machines, they can conduct experiments in the world and empirically test the information they were provided. This physical embodiment is often described as the missing piece for LLMs to truly understand the world.Bisk et al. (2020) Just like human cultural narratives can be shifted by government, policies, advertising, activism, and pop culture, artificial cultural narratives should become more malleable. Autotelic agents must be given the possibility

to steer, edit and extend their cultural models (i.e. LLMs) in light of their embodied experience; to share it and negotiate it with others; i.e. to participate in a shared cultural evolution. Reversely, humans that train language/cultural models should pay great care to build and understand the cultural input they provide, just like they pay great care to the education of their children.

Through this process, agents could learn some of the uniquely human social features described by Tomasello in his recent bookTomasello (2019): cooperative thinking, moral identity or social norms. However, a high-quality alignment may require humans to enter the interaction loop, enabling autotelic agents to ground, verify and correct the cultural knowledge they acquired with LLMs. Furthermore, mere exposure to culture (either through LLMs or direct interaction with humans) might not be sufficient to design truly autonomous social learners. This may require encoding certain mechanisms such as joint intentionality or other collaborative priors inside agents — the topic of social RL.Jaques et al. (2019)

Challenge #4: Pursuing long-term goals. Current autotelic agents mostly pursue goals at the timescale of an episode. Humans, on the other hand, can pursue goals they can barely hope to achieve within their lifetime (e.g. building an efficient fusion reactor). Because there is an infinity of potential goals and little time to explore them alone, autotelic agents may need cultural models to bias their selection of long-term goals towards more feasible, interesting, or valuable options — turning an individual exploration into a *population-based exploration*. Keeping long-term goals in mind will require improvements in architecture's memory systems, but might also benefit from language and culture. Indeed, verbalization is known to increase humans' memory span Elliott et al. (2021) and writing let us set our goals in stone (from the post-it note reminding you to take the garbage out to the Ten Commandments). Young children progressively become future-oriented as they are taught to project themselves into the future through education, social interactions (*what do you want to do when you grow up?*) and cultural metaphors (e.g. the self-made man).Atance (2008) If autotelic agents will need better hierarchical RL algorithms to achieve long-term goals, they could also leverage cultural artifacts evolved for improved collaboration and long-term planning — think of roadmaps, organization systems and project management tools.Clark (1998a) Because long-term goals are not rewarding, human cultures supply short-term social rewards (good grades in the educational system, money and social recognition in professional careers) — a form of reward shaping.

Chapter 10

Summary

Conclusion goes here.

Appendices

Appendix A

ABIG

This Supplementary Material provides additional derivations, implementation details and results. More specifically:

- Section A.1 proposes derivations, implementation details and analysis related to our method.
  - Subsection A.1.1 provides additional diagrams illustrating the ABP problem and its position with respect to related settings.
  - Subsection A.1.2 proposes the full derivation of the agents' MDP.
  - Subsection A.1.3 proposes our methods pseudo-code, algorithmic implementation details (for BC and MCTS), hyper-parameters and compute resources.
  - Subsection A.1.4 proposes analysis that explore our method's learning mechanisms.
  - Subsection A.1.5 discusses the differences between ABP and Hierarchical/Feudal Reinforcement Learning.
- Section A.2 provides additional results.
  - Subsection A.2.1 monitors the builder's behavior properties as training progresses.
  - Subsection A.2.2 compares our method to additional baselines.
  - Subsection A.2.3 analyses the impact of the vocabulary size on the learning performance.

A.1 Supplementary Methods

A.1.1 Supplementary Sketches

A.1.2 Analytical Description

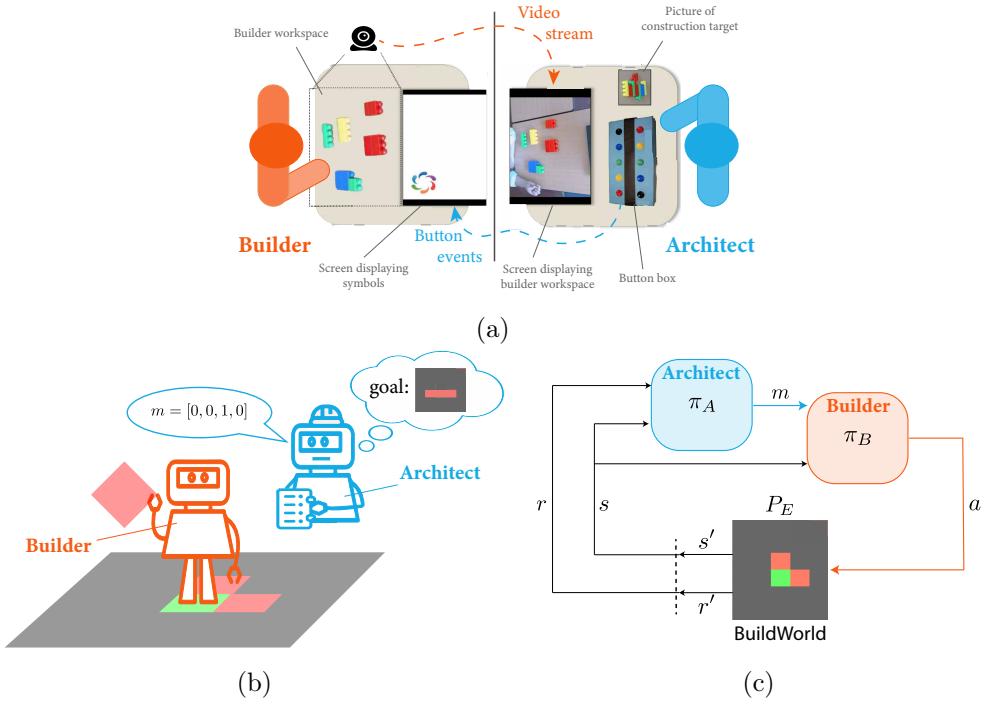


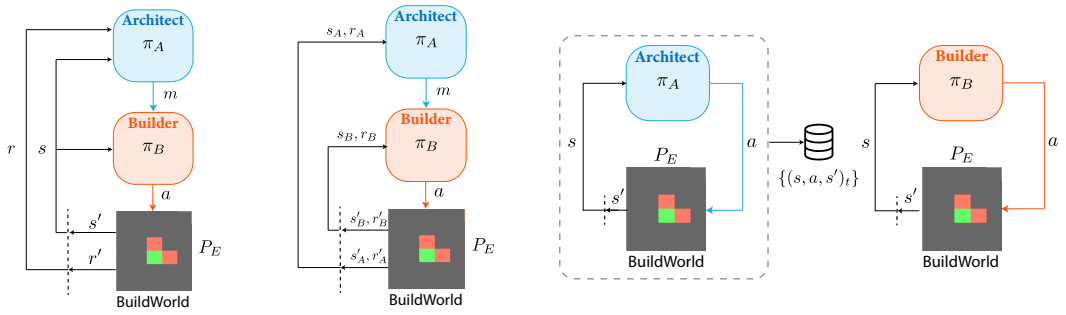
Figure A.1: (a) **Schematic view of the CoCo Game.** The architect and the builder should collaborate in order to build the construction target while located in different rooms. The architecture has a picture of the target while the builder has access to the blocks. The architect monitors the builder workspace via a camera (video stream) and can communicate with the builder only through the use of 10 symbols (button events). (b) **Schematic view of the Architect-Builder Problem.** The architect must learn how to use messages to guide the builder while the builder needs to learn to make sense of the messages in order to be guided by the architect. (c) **Interaction diagram between the agents and the environment in our proposed ABP.** The architect communicates messages (m) to the builder. Only the builder can act (a) in the environment. The builder conditions its action on the message sent by the builder ($\pi_B(a|s,m)$). The builder never perceives any reward from the environment

Transition Probabilites from the architect point of view

Using the laws of total probabilities and conditional probabilities we have:

$$\begin{aligned} P_A(s'|s,m) &= \sum_{a\in\mathcal{A}} P(s',a|s,m) \\ &= \sum_{a\in\mathcal{A}} P(s'|a,s,m)P(a|s,m) \\ &= \sum_{a\in\mathcal{A}} P_E(s'|a,s)\tilde{\pi}_b(a|s,m) \end{aligned} \tag{A.1}$$

Where the final equality uses the knowledge that next-states only depends on states and builder's actions.



(a) (b) (c)

Figure A.2: (a) **Vertical view of the interaction diagram between the agents and the environment in our proposed ABP.** Only the architect perceives a reward signal r; (b) **Interaction diagram for a standard MARL modelization.** Both the architect and the builder have access to environmental rewards r_A and r_B. Which would contradict the fact that the builder ignores everything about the task at hand; (c) **Inverse Reinforcement Learning modelization of the ABP.** The architect needs to provide demonstrations. The architect does not exchange messages with the builder. The builder relies on the demonstrations $\{(s, a, s')_t\}$ to learn the desired behavior.

Reward function from the architect point of view

$$
\begin{aligned}
r_{_A}(s, m, s') &\triangleq \mathbb{E}[R|s, m, s'] \\
&= \int_{\mathbb{R}} rP(r|s, m, s')dr \\
&= \int_{\mathbb{R}} r \sum_{a \in \mathcal{A}} P(r, a|s, m, s')dr \\
&= \int_{\mathbb{R}} r \sum_{a \in \mathcal{A}} P(r|s, m, a, s')P(a|s, m, s')dr \\
&= \int_{\mathbb{R}} r \sum_{a \in \mathcal{A}} P(r|s, a, s')\tilde{\pi}_b(a|s, m)dr \\
&= \sum_{a \in \mathcal{A}} \tilde{\pi}_b(a|s, m) \int_{\mathbb{R}} rP(r|s, a, s')dr \\
&= \sum_{a \in \mathcal{A}} \tilde{\pi}_b(a|s, m)r(s, a, s')
\end{aligned}
\tag{A.2}
$$

Transition function from the builder point of view

$$
\begin{aligned}
P(s', m'|s, m, a) &= P(m'|s', s, m, a)P(s'|s, m, a) \\
&= P(m'|s')P(s'|s, a) \\
&= \tilde{\pi}_{_A}(m'|s')P_{_E}(s'|s, a)
\end{aligned}
\tag{A.3}
$$

A.1.3 Practical Algorithm

Algorithm 2: Architect-Builder Iterated Guiding (ABIG)

Require: randomly initialized builder policy π_B, reward function r, transition function P_E, BC algorithm, MCTS algorithm
for i in range($N_{iterations}$) **do**
  MODELLING FRAME:
    for e in range($N_{collect}/2$) **do**
      Architect populates $\mathcal{D}_A$ using $m \sim$ Uniform() and observing $a \sim \pi_B(\cdot|s, m)$
    end for
    Architect learns $\tilde{\pi}_B(a|s, m)$ on $\mathcal{D}_A$ with BC
    Architect sets $\pi_A(m|s) \triangleq$ MCTS$(r, \tilde{\pi}_B, P_E)$
    Architect flushes $\mathcal{D}_A$
  GUIDING FRAME:
    for e in range($N_{collect}/2$) **do**
      Builder populates $\mathcal{D}_B$ using π_B while guided by Architect, i.e. $m \sim \pi_A(\cdot|s)$
    end for
    Builder learns $\pi_B(a|s, m)$ on $\mathcal{D}_B$ with BC
    Builder flushes $\mathcal{D}_B$
end for
Architect runs one last Modelling Frame
Result: π_A, π_B

Behavioral Cloning

The data-set is split into training (70%) and validation (30%) sets. If the validation accuracy does not improve during a *wait for* number of epochs the training is early stopped. For a training data-set $\mathcal{D} = \{(s, m, a)\}$ of size N the BC loss to minimize for a policy π_θ parametrized by θ is given by:

$$J(\theta) = \frac{1}{N} \sum_{\mathcal{D}} -\log \pi_\theta(a|s, m) \tag{A.4}$$

Monte-Carlo Tree Search

In the architect's MCTS, nodes are labeled by environment's states and they are expanded by selecting messages. Selecting message m from a node with label s yields a builder action according to the architect's builder model $a \sim \tilde{\pi}_B(a|s, m)$, this sampled action in turn yields the label of the child node according to the environment's transition model $s' \sim P_E(s'|s, a)$. We repeat this process until we select a message that was never selected from the current node or we sample a next state that does not correspond to a child node yet. In both of these cases a new node has to be created. We estimate the value of the new node using an engineered heuristic that estimates the return of an optimal policy $\pi^*(a|s)$ from state s. This value is scaled down by a factor 2 to avoid

overestimation: the builder's policy may not allow the architect to have it follow π^*. This estimated value for a newly created node at depth l is back-propagated as a return to parents node at depth k according to:

$$G^k = \sum_{\tau=0}^{l-1-k} \gamma^\tau r_{k+1+\tau} + \gamma^{l-k} v^l \qquad k = l, ..., 0 \tag{A.5}$$

where r_j is the reward collected from node at depth j to child node at depth $j+1$. From a node with label s we select messages according to the Upper Confidence Bound rule:

$$\begin{aligned} m &= \operatorname*{argmax}_m Q(s,m) + c\sqrt{\frac{\ln \sum_b N(s,b)}{N(s,m)}} \\ Q(s,m) &= \frac{\sum_i G_i(s,m)}{N(s,m)} \end{aligned} \tag{A.6}$$

where $N(s,m)$ is the number of times message m was selected from the node, $G_i(s,m)$ are the returns obtained from the node when selecting m and c is a constant set to $\sqrt{2}$. When the architect must choose a message from the environment state s, its policy $\pi_{_A}(m|s)$ runs the above procedure from a root node labeled with the current environment state s. After expanding a budget b of nodes the architect picks the best message to send according to Eq. (A.6) applied to the root node. It is then possible to reuse the tree for the next action selection or to discard it, if a tree is reused its maximal depth should be constrained.

Hyper-parameters

sampling temperature	samples per iteration	learning rate	number of epochs	batch size
0.5	100	0.1	1000	50

Table A.1: Toy experiment hyper-parameters

budget	reuse tree	max tree depth
100	true	500

Table A.2: MCTS parameters

Sparse reward means that the architect receives 1 if the goal is achieved and 0 otherwise. Episodes per iterations are equally divided into the modelling and guiding frames. Only the learning rates on BuildWorld were searched over with grid-searches. For BuildWorld with 3 blocks the searched range is $[5\times10^{-4}, 1\times10^{-4}, 1\times10^{-5}]$ for both architect and builder (vocabulary size was fixed at 6). For 'grasp' with 6 blocks the searched range is $[1\times10^{-3}, 5\times10^{-4}, 1\times10^{-4}]$ for the architect and $[5\times10^{-4}, 1\times10^{-4}, 5\times10^{-5}]$ for the

episode len	grid size	reward	message
40	5×6 / (6 × 6)	sparse	one-hot

discount factor	episodes per iteration	vocab size	evaluation episode len
0.95	600	18 / (72)	40 / (60)

Table A.3: BuildWorld parameters for 3 blocks / (for 6 blocks if different)

learning rate	number of epochs	batch-size	wait for
5×10^{-4}	1000	256	300

Table A.4: Architect's BC parameters on BuildWorld for 3 blocks / (for 6 blocks if different)

learning rate	number of epochs	batch-size	wait for
1×10^{-4}	1000	256	300

Table A.5: Builder's BC parameters on BuildWorld for 3 blocks / (for 6 blocks if different)

builder (vocabulary size was fixed at 72). The other hyper-parameters do not seem to have a major impact on the performance provided that:

- the MCTS hyper-parameters enable an agent that has access to the reward to solve the task.
- there is enough BC epochs to approach convergence.

Regarding the vocabulary size, the bigger the better (see experiments in Figure A.8).

Computing resources

A complete ABIG training can take up to 48 hours on a single modern CPU (`Intel E5-2683 v4 Broadwell @ 2.1GHz`). The presented results require approximately 700 CPU hours. For each training, the main computation cost comes from the MCTS planning during the guiding frames. The self-imitation and behavior modelling steps only account for a small fraction of the computation.

A.1.4 Intuitive Explanation of the Learning Dynamics

To illustrate the learning mechanisms of ABIG we propose to look at the simplest instantiation of the Architect-Builder Problem: there is one state (thus it can be ignored), two messages m_1 and m_2 and two possible actions a_1 and a_2. If the builder chooses a_1 it is a loss ($r(a_1) = -1$) but choosing a_2 results in a win ($r(a_2) = 1$). Figure A.3 displays several iterations of ABIG on this problem when the initial builder's policy is unfavorable (a_1 is more likely than a_2 for all the messages). During each iteration the architect selects messages in order to maximize the likelihood of the builder picking action a_2 and then the

builder does self-Imitation Learning by maximizing the likelihood of the corresponding messages-actions sequence under its policy. Figure A.3 shows that this process leads to forgetting unfavorable associations until a favorable association emerges and can be reinforced. On the other hand, for ABIG-no-intent in Figure A.4, favorable and unfavorable messages are sampled alike which prevents the forget mechanism to undo unfavorable message-to-action associations. Consequently, initial preferences are reinforced.

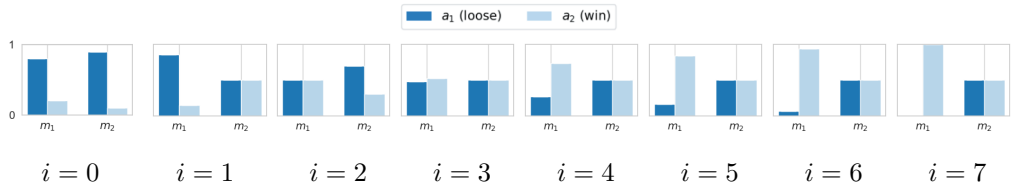


Figure A.3: ABIG-driven evolution of message-conditioned action probabilities (builder's policy) for a simple problem where the builder must learn to produce action a_2. Even under unfavorable initial condition the architect-builder pair eventually manages to associate a message (here m_1) with the winning action (a_2). Initial conditions are unfavorable since a_1 is more likely than a_2 for both messages. ($i = 0$) Given the initial conditions, the architect only sends message m_1 since it is the most likely to result in action a_2. ($i = 1$) the builder guiding data only consisted of m_1 message therefore it cannot learn a preference over actions for m_2 and both actions are equally likely under m_2. The architect now only sends message m_2 since it is more likely than m_1 at triggering a_2. ($i = 2$) Unfortunately, the sampling of m_1 resulted in the builder doing more a_1 than a_2 during the guiding frame and the builder thus associates m_2 with a_1. The architect tries its luck again but now with m_1. ($i = 3$) Eventually, the sampling results in more a_2 actions being sampled in the guiding data and the builder now associates m_1 to a_2. ($i = 4$) and ($i = 5$) The architect can now keep on sending m_1 messages to reinforce this association.

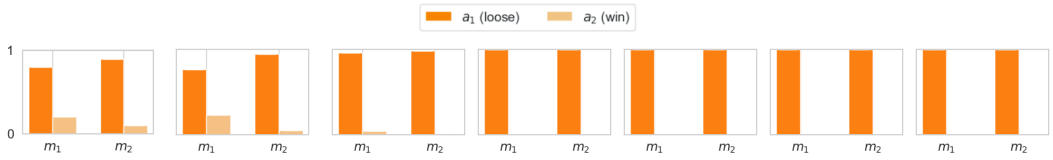


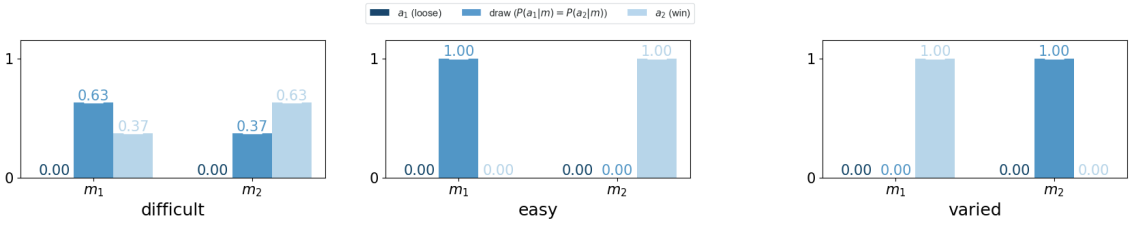
Figure A.4: ABIG-no-intent driven evolution of message-conditioned action probabilities for a simple problem where builder must learn to produce action a_2. Initial conditions are unfavorable since a_1 is more likely than a_2 for both messages. Without an architect's guiding messages during training, a self-imitating builder reinforces the action preferences of the initial conditions and fails (even when evaluated alongside a knowledgeable architect as both messages can only yield a_1).

To further assess how the architect's message choices impact the performance of a self-imitating builder, we compare the distribution of the builder's preferred actions obtained after using ABIG and ABIG-no-intent. We consider three different initial conditions (favorable, unfavorable, intermediate) that are each ran to convergence (meaning that the policy does not change anymore across iterations) for 100 different seeds. Figure A.5

displays the resulting distributions of preferred – i.e. most likely – action for each message. When applying ABIG on the toy problem, the pair always reaches a success rate of 100/100 no matter the initial condition. We also observe that – at convergence – the builder never prefers action a_1, yet when an action is preferred for a given message, the other message yields no preference over action ($p(a_1|m) = p(a_2|m)$). This is due to the forget mechanism discussed in Section 4.3.3. The results when applying ABIG-no-intent on the toy problem are much more dependent on the initial condition. In the unfavorable scenario, ABIG-no-intent fails heavily with only 3 seeds succeeding over the 100 experiments. This is due to the fact that, in absence of message guidance from the architect, the builder has high chances to continually reinforce the association between the two messages and a_1, therefore losing. However, in rare cases, the builder can inverse the initial message-conditioned probabilities by 'luckily' sampling more often a_2 when receiving m_1 and win. This only happened 3 times over the 100 seeds. Finally, when initial conditions are more favorable, the self-imitation steps reinforce the association between the messages and a_2 which makes the builder prefer a_2 for at least one message and enables high success rates (100/100 for favorable and 98/100 for intermediate).

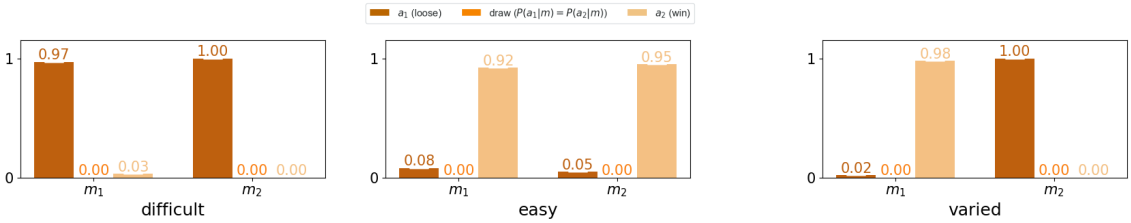
Unfavorable	Favorable	Intermediate
(a) **Initial probabilities**		
$P(a_1\|m_1) = 0.8$, $P(a_2\|m_1) = 0.2$ $P(a_1\|m_2) = 0.9$, $P(a_2\|m_2) = 0.1$	$P(a_1\|m_1) = 0.2$, $P(a_2\|m_1) = 0.8$ $P(a_1\|m_2) = 0.1$, $P(a_2\|m_2) = 0.9$	$P(a_1\|m_1) = 0.9$, $P(a_2\|m_1) = 0.1$ $P(a_1\|m_2) = 0.1$, $P(a_2\|m_2) = 0.9$

(b) **ABIG:** Distributions of final preferred action for each message calculated over 100 seeds



Success Rate = 100/100 | Success Rate = 100/100 | Success Rate = 100/100

(c) **ABIG-no-intent:** Distributions of final preferred action for each message calculated over 100 seeds



Success Rate = 3/100 | Success Rate = 100/100 | Success Rate = 98/100

Figure A.5: **Toy experiment analysis** (a) Initial conditions: initial probability for each action a given a message m; distributions of final builder's preferred actions for each message after applying (b) ABIG and (c) ABIG-no-intent on the toy problem; distributions are calculated over 100 seeds. For each method and each initial condition, we report the success rate obtained by a knowledgeable architect guiding the builder. At evaluation, the architect has access to the builder's model and does not ignore the goal. ABIG always succeeds while ABIG-no-intent's success depends on the initial conditions.

A.1.5 Related Work

In this section we develop the differences between ABP and Hierarchical/Feudal Reinforcement Learning more in detail.

Kulkarni et al. (2016) proposes to decompose a RL agent into a two-stage hierarchy with a meta-controller (or manager) setting the goals of a controller (or worker). The meta-controller is trained to select sequences of goals that maximize the environment reward while the controller is trained to maximize goal-conditioned intrinsic rewards. The definition of the goal-space as well as the corresponding hard-coded goal-conditioned reward functions are task-related design choices. In Vezhnevets et al. (2017), the authors propose a more general approach by defining goals as embeddings that directly modulate the worker's policy. Additionally, the authors define intrinsic rewards as the cosine

distance between goals and embedded-state deltas (difference between the embedded-state at the moment the goal was given and the current embedded-state). Thus, goals can be interpreted as directions in embedding space. Nachum et al. (2018) build on a this idea but let go of the embedding transformation by considering goals as directions to reach and rewards as distances between state deltas and goals. These works tackle the single-agent learning problem and therefore allow the manager to directly influence the learning signal of the workers. However, in the multi-agent setting where agents are physically distinct, it is not possible for an agent to explicitly tweak another agent's learning algorithm. Instead, agents must communicate by influencing each other's observations instead of intrinsic rewards. Since it is designed to investigate the emergence of communication between agents, ABP lies in this latter multi-agent setting where agents can interact with one-another only through observations. This makes applying Feudal or Hierarchical methods to the ABP unfeasible as they are restricted to worker agents that directly receive rewards. In contrast, in ABP, the reward-less builder observes communication messages that, initially, have arbitrary meaning.

A.2 Supplementary Results

A.2.1 Learning Analysis

The main document shows performances (success rates) of builder-architect pairs at convergence. In this supplementary section we propose to thoroughly study the evolution of the builder's policy in order to provide a deeper analysis of ABIG.

Metric definition. We define three metrics that characterize the builder behavior. We compute these metrics on a constant *Measurement Set* $\mathcal{M}$ made of 6000 randomly sampled states, for each of these states we sample all the possible messages $m \sim \text{Uniform}(\mathcal{V})$ where $\mathcal{V}$ is the set of possible messages. Therefore, $|\mathcal{M}| = 6000 \times |\mathcal{V}|$. The set of possible actions is $\mathcal{A}$ and we denote by δ the indicator function.

We also define the following distributions:

$$
\begin{aligned}
&p_{S}(s) \triangleq \frac{1}{|\mathcal{M}|} \sum_{s' \in \mathcal{M}} \delta(s' == s) \\
&p_{M}(m) \triangleq P(m|s) = \frac{1}{|\mathcal{V}|} \\
&p_{SM}(s,m) \triangleq p_{S}(s)P(m|s) = p_{S}(s)p_{M}(m) \\
&p_{SMA}(s,m,a) \triangleq p_{SM}(s,m)P(a|s,m) = p_{SM}(s,m)\pi_{B}(a|s,m) \\
&p_{A}(a) \triangleq \sum_{(s,m) \in \mathcal{M}} p_{SMA}(s,m,a) \\
&p_{MA}(m,a) \triangleq \sum_{s \in \mathcal{M}} p_{SMA}(s,m,a) \\
&p_{SA}(s,a) \triangleq \sum_{m \in \mathcal{M}} p_{SMA}(s,m,a)
\end{aligned}
$$

From this we can define the monitoring metrics:

- *Mean Entropy:*

$$\bar{H}(\pi) = \frac{1}{|\mathcal{M}|} \sum_{(s,m) \in \mathcal{M}} \left[- \sum_{a \in \mathcal{A}} \pi(a|s,m) \log \pi(a|s,m) \right]$$

- *Mutual Information between messages and actions*

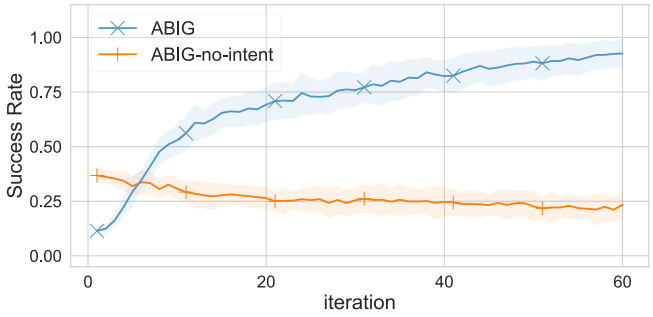
$$I_m = \sum_{m \in \mathcal{V}} \sum_{a \in \mathcal{A}} p_{MA}(m,a) \log \frac{p_{MA}(m,a)}{p_{A}(a)p_{M}(m)}$$

- *Mutual Information between states and actions*

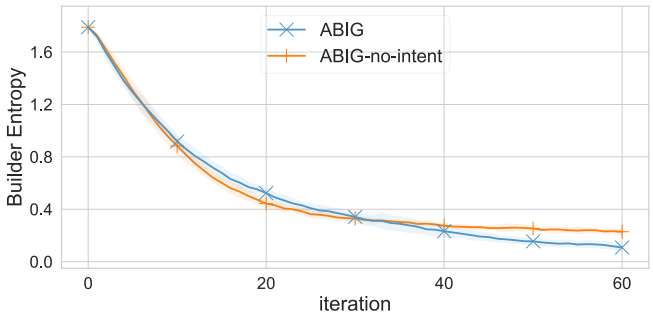
$$I_s = \sum_{s \in \mathcal{M}} \sum_{a \in \mathcal{A}} p_{SA}(s,a) \log \frac{p_{SA}(s,a)}{p_{A}(a)p_{S}(s)}$$

Analysis. Figure A.6 displays the evolution of these metrics after each iteration as well as the evolution of the success rate (a). As indicated by Eq. (4.5), doing self-imitation learning results in a decay of the mean entropy (b). This decay is similar for

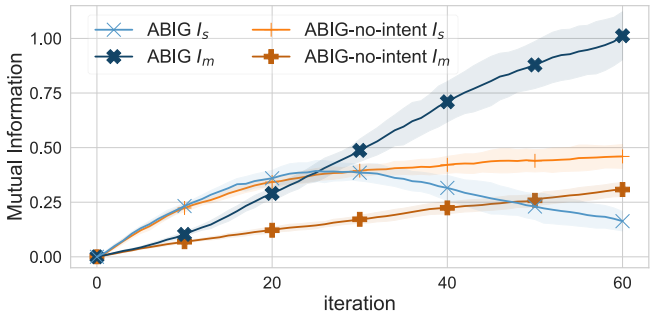
ABIG and ABIG-no-intent. The most interesting result is provided by the evolution of the mutual information (c). For ABIG-no-intent, we see that I_s and I_m slowly increase with $I_s > I_m$ over all iterations. This indicates that the builder policy $\pi_B(a|s,m)$ relies more on states than on messages to compute the actions. In this scenario the builder, therefore, tends to ignore messages. On the other hand, I_s and I_m evolve differently for ABIG. Both metrics first increase with $I_s > I_m$ until they cross around iteration 25. Then I_s starts decreasing and I_m grows. This shows that ABIG results in a builder policy that strongly selects actions based on the messages it receives which is a desirable feature of emergent communication.



(a) Evolution of the success rate



(b) Evolution of the builder policy mean entropy $\bar{H}_{\pi_B}$



(c) Evolution of the mutual information I_s and I_m

Figure A.6: Comparison of the evolution of builder policy properties when applying ABIG and ABIG-no-intent on the 'place' task in BuildWorld. (a) ABIG enables much higher performance that ABIG-no-intent. (b) Both methods use self-imitation and thus reduce the entropy of the policy. (c) ABIG promotes the mutual information between messages and action which indicates successful communication protocols.

A.2.2 Additional Baseline Comparison

We define two extra baselines:

- Stochastic: where the builder policy is a fixed softmax policy parameterized by a randomly initialized network;
- Deterministic: where the builder policy is a fixed argmax policy parameterized by a randomly initialized network.

In the performances reported in Figure A.7, the architect has direct access to the exact policy of the builder ($\tilde{\pi}_B = \pi_B$) and uses it to plan and guide the builder during evaluation. We observe that the stochastic condition exhibits similar performances as the random builder. This indicates that, even if the architect tries to guide the builder, the stochastic policy is not controllable and performances are not improved. Finally, we would expect a deterministic policy to be more easily controllable by the architect. Yet, as pointed out in Figure A.7, the initial deterministic policies lack flexibility and fail. This shows that the builder must iteratively evolve its policy in order to make it controllable.

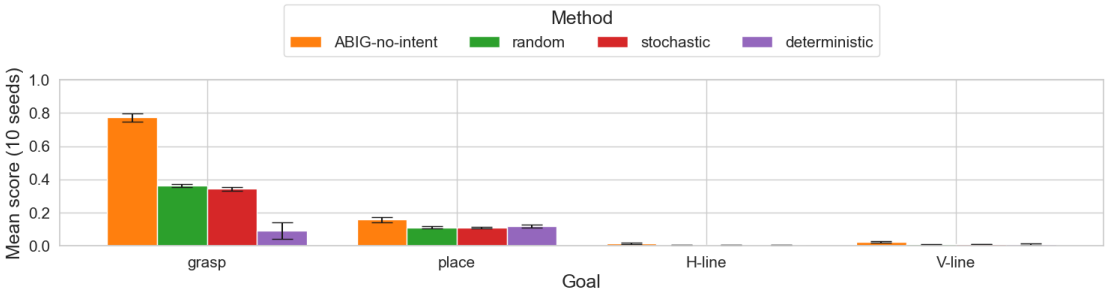


Figure A.7: Baseline performance depending on the goal: stochastic policy behaves on par with random builder. Self-imitation with ABIG-no-intent remains the most controllable baseline.

A.2.3 Impact of the Vocabulary Size

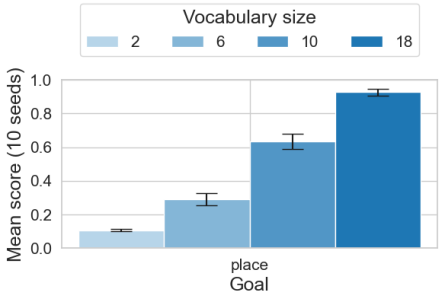


Figure A.8: Influence of the Vocabulary size for ABIG on the 'place' task. Performance increases with the vocabulary size.

Appendix B

CURVES

This Supplementary Material provides additional derivations, implementation details and results. More specifically:

- Section A provides supplementary implementation details in the form of:
  - Images of testing set of visual referents;
  - Topographic score derivation;
  - Training procedures and hyperparameters;
  - Pseudo-code.
- Section B provides supplementary results:
  - Auto-comprehension generalization performances;
  - Additional Lexicons;
  - Utterances examples across perspectives illustrating coherence;
  - Topographic maps & scores;
  - Composition matrix examples;
  - T-SNEs of embeddings;

B.1 Supplementary Methods

B.1.1 Testing Set

Figure B.1: **Perspective instances of the testing set $\mathcal{R}_5^2$.**

B.1.2 Topographic Score

To evaluate the compositionality of the emerging language we define the topographic score:

$$\rho_{ij} = ||(O, h_{ij})||_2 - ||(O, h_k)||_2 \text{ with } k = \text{argmin}_{k \in \{i,j\}} ||h_k, h_{ij}||_2) \tag{B.1}$$

It is obtained by computing the Hausdorff distance between the utterances denoting compositional referents with respect to both the utterance denoting the single feature i $(d_H(u(r_i), \cdot))$and the one denoting the single feature j $(d_H(u(r_j), \cdot))$. To derive our metric, we define 4 groups of utterances denoting compositional referents.

- $u(r_{ij})$ the utterances for referent made of feature i and j.
- $u(r_{xj}, x \neq i)$ the utterances denoting referent made by composing feature j with any other feature different than i
- $u(r_{iy}, y \neq j)$ the utterances denoting referent made by composing feature i with any other feature different than j
- $u(r_{xy})$ the utterance denoting all other compositional referents in $\mathcal{R}_5^2$.

and compute their Hausdorff distances to $u(r_i)$ and $u(r_j)$. As displayed in Figure B.2, if utterances $u(r_{ij})$ are compositional we expect them to be at the same time close to $u(r_i)$ and close to $u(r_j)$ and hence to land in the bottom left corner of the distance graph. Moreover, they should be closer to the origin than $u(r_{xj})$ and $u(r_{iy})$. To quantify to what extent it is the case we compute the barycenter of each group h_i, h_j, h_{ij} and h_{xy} and compute "how closer to the origin" is the compositional barycenter h_{ij} compared to its closest barycenter using equation B.1.

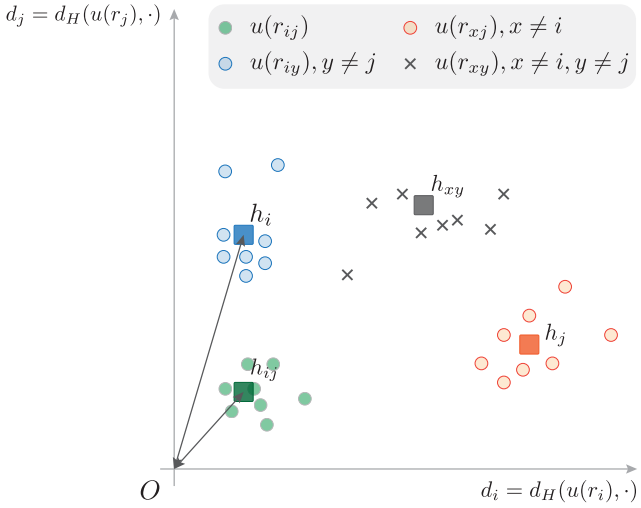


Figure B.2: Idealized mapping of utterances denoting compositional referents in the plan representing distances to utterances naming isolated features i and j.

B.1.3 Training procedure and hyperparameters

Agents have two separate encoders based on the same model architecture described in Table.B.1. Each agent performs association updates with a single step of gradient descent, using its own Adam optimizer with a learning rate of $1e^{-4}$. To allow faster convergence, agents perform an association update between an abstract referent $r_A^\star$ and an utterance u by using a batch of 64 perspectives $\{\Phi(r_A^\star)\}_{i\in[1,64]}$. From a cognitive science perspective, this is comparable to an agent "walking around" an object to better understand how different perceptions relate to the same object. From a computer science perspective, this is similar to the self-supervised framework of SimCLR (Chen et al., 2020), where agents learn representation by contrastively aligning the embeddings of an input with these of the same transformed input.

Layer	Activation
Conv2D(filters=8, stride=2, padding=1)	ReLU
Conv2D(filters=16, stride=2, padding=1)	ReLU
Conv2D(filters=32, stride=2, padding=0)	ReLU
Linear(128)	ReLU
Linear(32)	None

Table B.1: **Model architecture used for both the referent and utterance Encoders.** (when referents are one-hot vectors, the 3 Conv2D layers are replaced by a Linear layer with ReLu activation)

While the drawing pipeline is fully differentiable, it is highly sensitive to local minima. Thus, we solve equation 5.5 in the descriptive case or equation 5.6 in the discriminative scenario by simultaneously performing gradient descent on a batch of 64 randomly initialized command vectors over 100 iterations, using a newly initialized Adam optimizer each time with a learning rate of $1e^{-2}$.

B.1.4 Pseudo-code

Algorithm 3: Speaker's Utterances

Require: perceived referents $\tilde{R}_S$, speaker's referent encoder f_S, speaker's utterance encoder g_S, sensory-motor system M

$Z_r \leftarrow f_S(\tilde{R}_S)$

$c \sim \text{Uniform}()$

for i in range($N_{\text{production}}$) **do**

$U_S \leftarrow M(c)$

$Z_u \leftarrow g_S(U)$

$S \leftarrow \text{sim}_{\cos}(Z_r, Z_u)$

$\mathcal{L} \leftarrow \text{mean}(\text{diag(S)}) * (-1)$

GD step on c to minimize $\mathcal{L}$

end for

Return $M(c)$

Algorithm 4: Listener's Selections & Binary Outcomes

Require: perceived referents $\tilde{R}_L$, produced utterances U_S, listener's referent encoder f_L, listener's utterance encoder g_L

$Z_r \leftarrow f_L(\tilde{R}_L)$

$Z_u \leftarrow g_L(U_S)$

$S \leftarrow \text{sim}_{\cos}(Z_r, Z_u)$

$t \leftarrow \text{argmax}(S, \text{axis}{=}1)$

$o \leftarrow \mathbf{0}$

for i in range($N_{\text{referents}}$) **do**

$o_i \leftarrow \mathbb{1}_{[t_i=i]}$

end for

Return o

Algorithm 5: Agents's Association Losses

Require: perceived referents $\tilde{R}_A$, produced utterances U_A, outcomes o, agent's referent encoder f_A, agent's utterance encoder g_A

$Z_r \leftarrow f_A(\tilde{R}_A)$
$Z_u \leftarrow g_A(U_A)$
$S \leftarrow \text{sim}_{\cos}(Z_r, Z_u)$
$\mathcal{L}_0 \leftarrow CE(S, \text{reduction=False})$
$\mathcal{L}_1 \leftarrow CE(S^\top, \text{reduction=False})$
$\mathcal{L} \leftarrow (\mathcal{L}_0 + \mathcal{L}_1)/2$
if $A =$ "S" **then**
  $\mathcal{L} \leftarrow (\mathcal{L} \cdot o)/N_{\text{referents}}$
  ENDIFELSE
  $\mathcal{L} \leftarrow (\mathcal{L} \cdot \mathbf{1})/N_{\text{referents}}$
end if
Return $\mathcal{L}$

B.2 Supplementary Results

B.2.1 Auto-comprehension generalization performances

Ref.	Auto	Social
One-hot	0.997 ± 0.005	0.991 ± 0.015
Visual-shared	0.862 ± 0.034	0.559 ± 0.027
Visual-unshared	0.425 ± 0.016	0.388 ± 0.02

Table B.2: Descriptive Success Rate

Ref.	Auto	Social
One-hot	0.997 ± 0.005	0.992 ± 0.009
Visual-shared	0.812 ± 0.019	0.567 ± 0.034
Visual-unshared	0.466 ± 0.019	0.404 ± 0.019

Table B.3: Descriminative Success Rate

We define the **Auto** performance metric as the communicative success rate, on test set, for language games involving a single agent playing as both the speaker and listener. We compare **Auto** and **Social** performances (the latter involving pairs of different agents, as done until now) in Tables B.3 & B.2.

B.2.2 Additional Lexicons

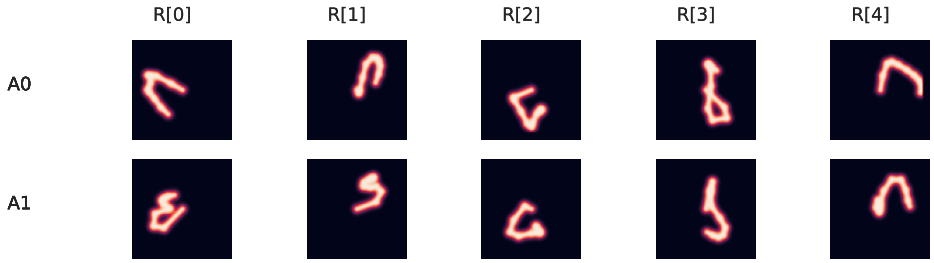


Figure B.3: **Instance of an emerging lexicon.** (Visual-shared).

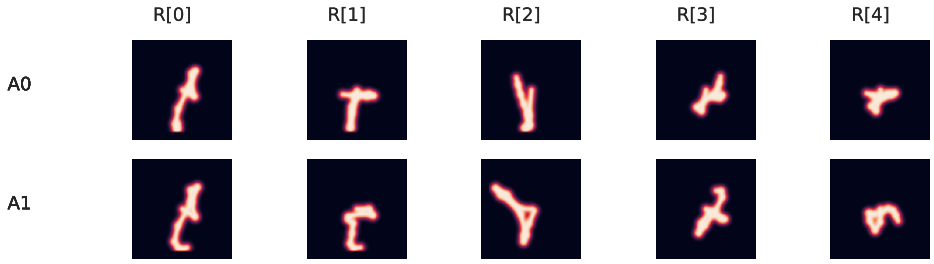


Figure B.4: **Instance of an emerging lexicon.** (One-hot).

B.2.3 Utterances examples across perspectives illustrating coherence.

The following figures illustrate the P-coherence and A-coherence of an emerging lexicon (Visual-unshared) by displaying, for each referent in R_1, the descriptive utterance produced for 10 random perspectives.

Figure B.5: Utterances examples for referent 0.

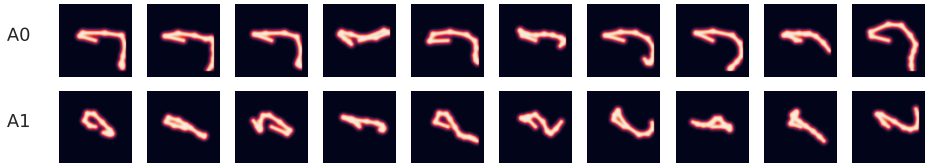


Figure B.6: Utterances examples for referent 1.

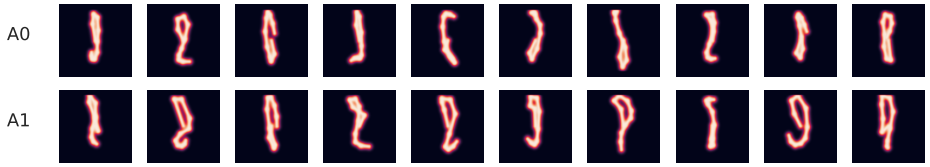


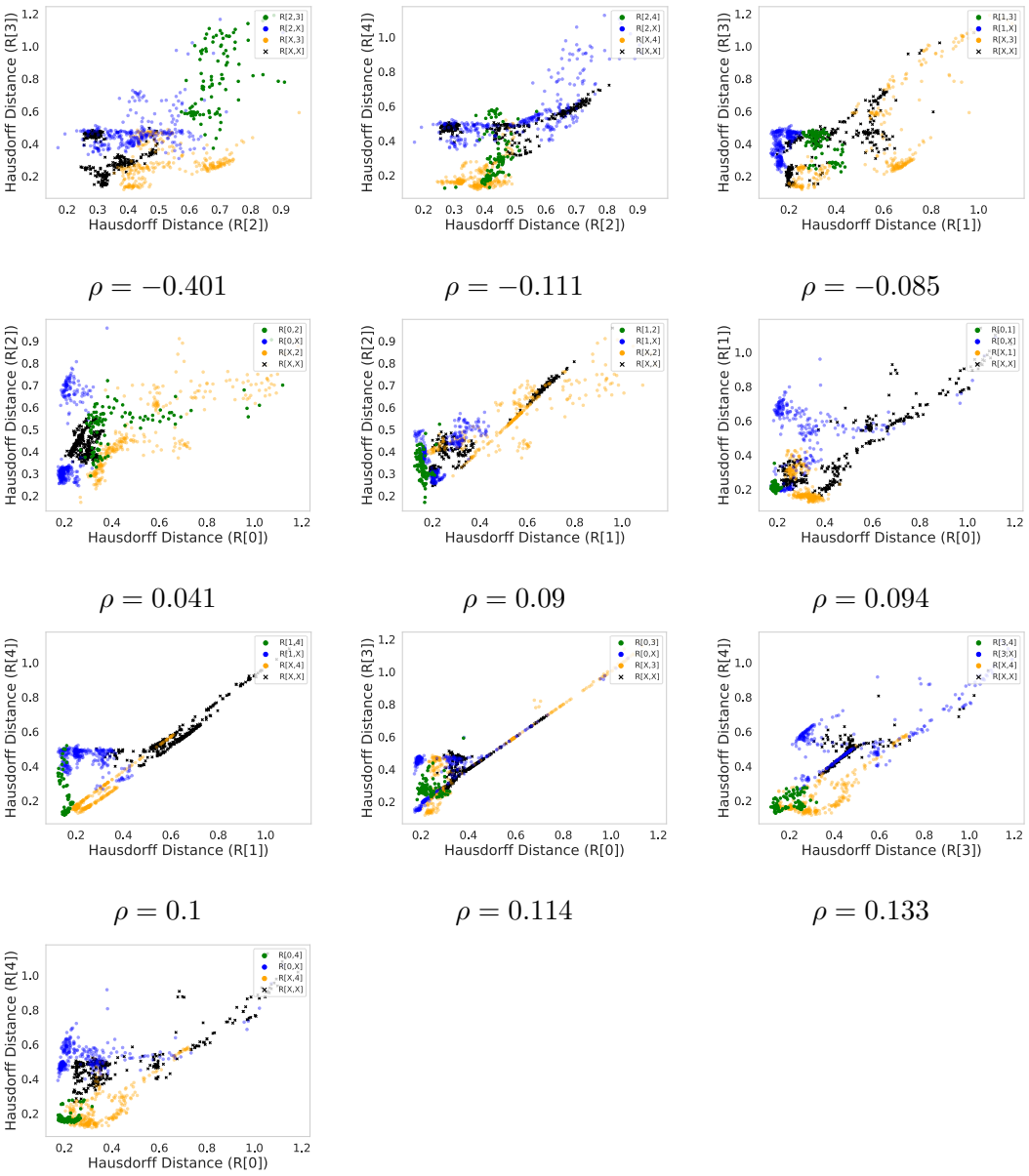
Figure B.7: Utterances examples for referent 2.

Figure B.8: Utterances examples for referent 3.

Figure B.9: Utterances examples for referent 4.

B.2.4 Topographic Maps & Scores

One-Hot



$\rho = -0.401$ $\rho = -0.111$ $\rho = -0.085$

$\rho = 0.041$ $\rho = 0.09$ $\rho = 0.094$

$\rho = 0.1$ $\rho = 0.114$ $\rho = 0.133$

$\rho = 0.147$

Figure B.10: Topographic maps and their associated topographic scores for each combination of features with one-hot referents

Visual - Shared Perspectives

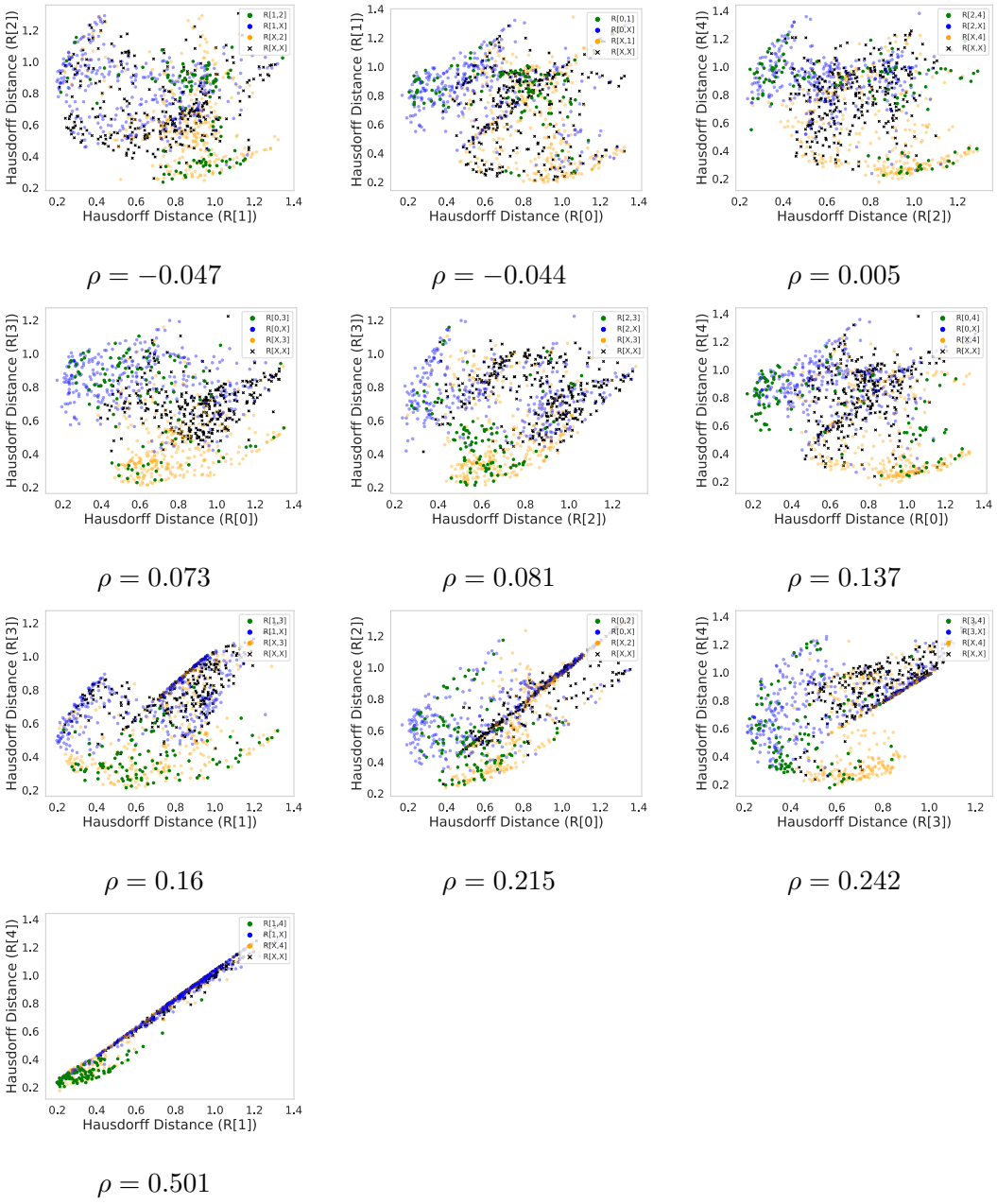


Figure B.11: Topographic maps and their associated topographic scores for each combination of features with shared-visual referents

Visual - Unshared Perspectives

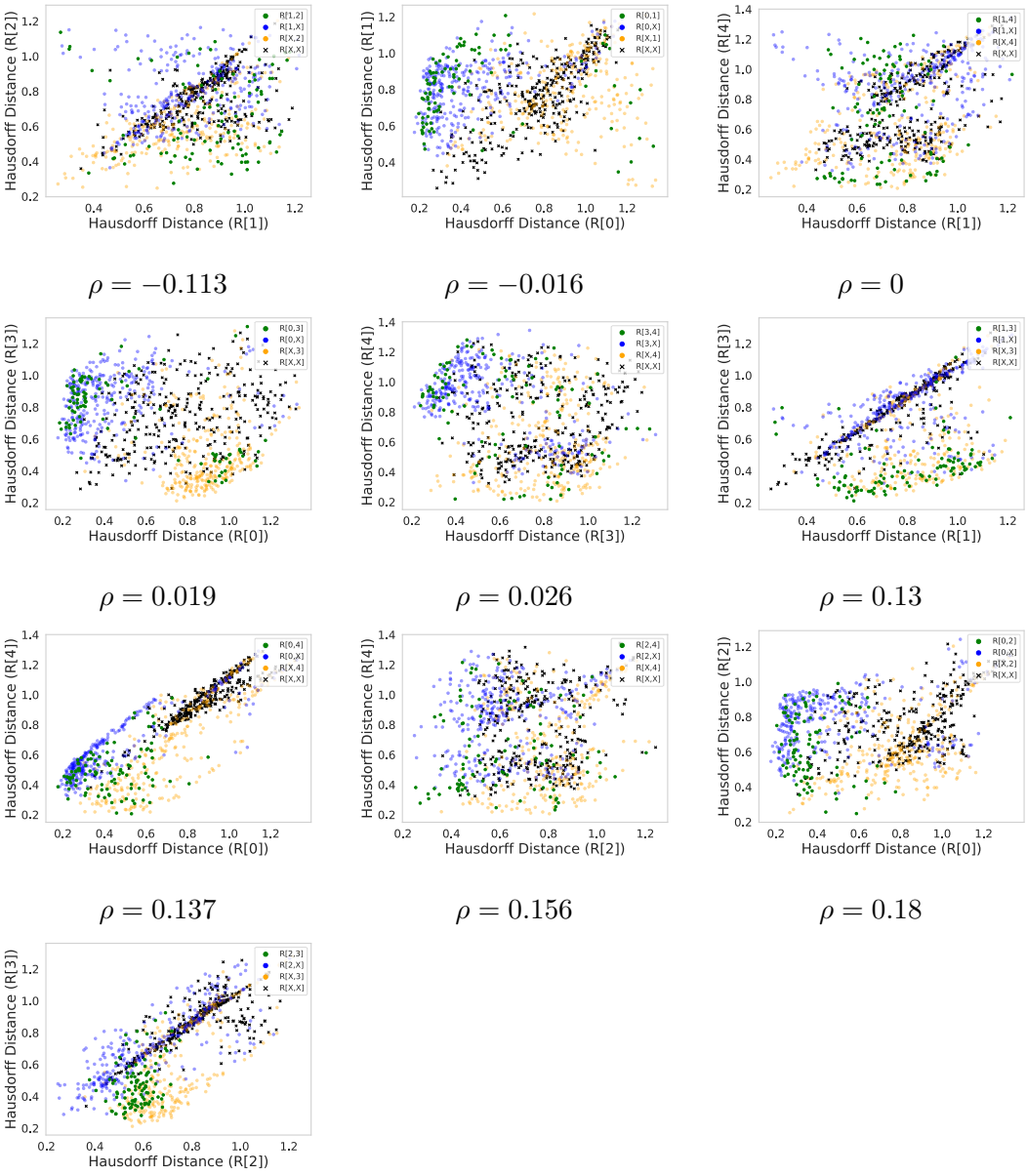


Figure B.12: Topographic maps and their associated topographic scores for each combination of features with unshared-visual referents

B.2.5 Composition Matrix examples (Visual - Unshared Perspectives)

Figure B.13: Instances of descriptive utterances for referents from R_1 (blue frames) and R_2.

B.2.6 T-SNEs of embeddings (Visual - Unshared Perspectives)

R_2 referents & descriptive utterances

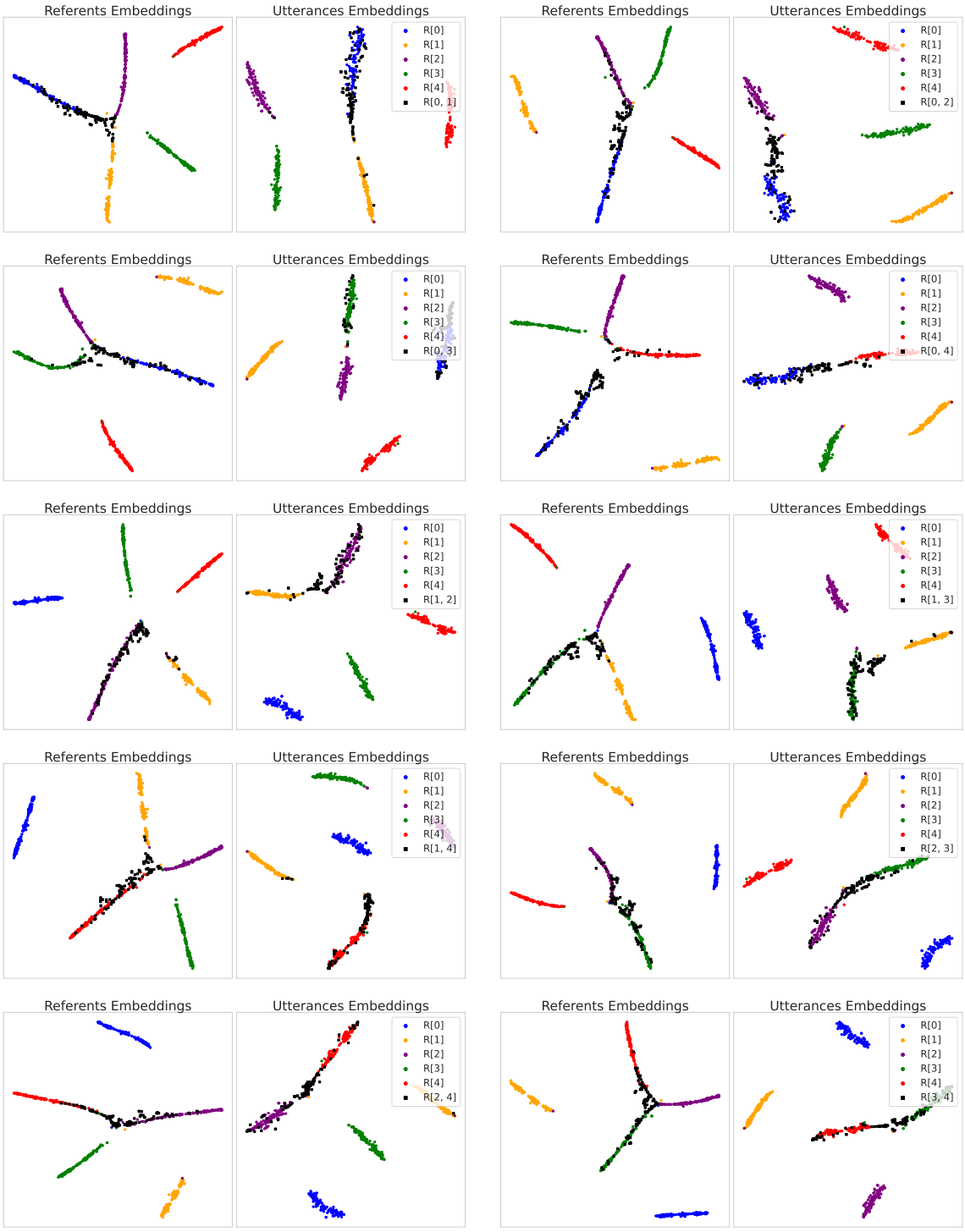


Figure B.14: **T-sne of referent and descriptive utterance embeddings.** Embeddings are computed for 100 perspectives of referents from R_2. Training conditions are unshared visual referents.

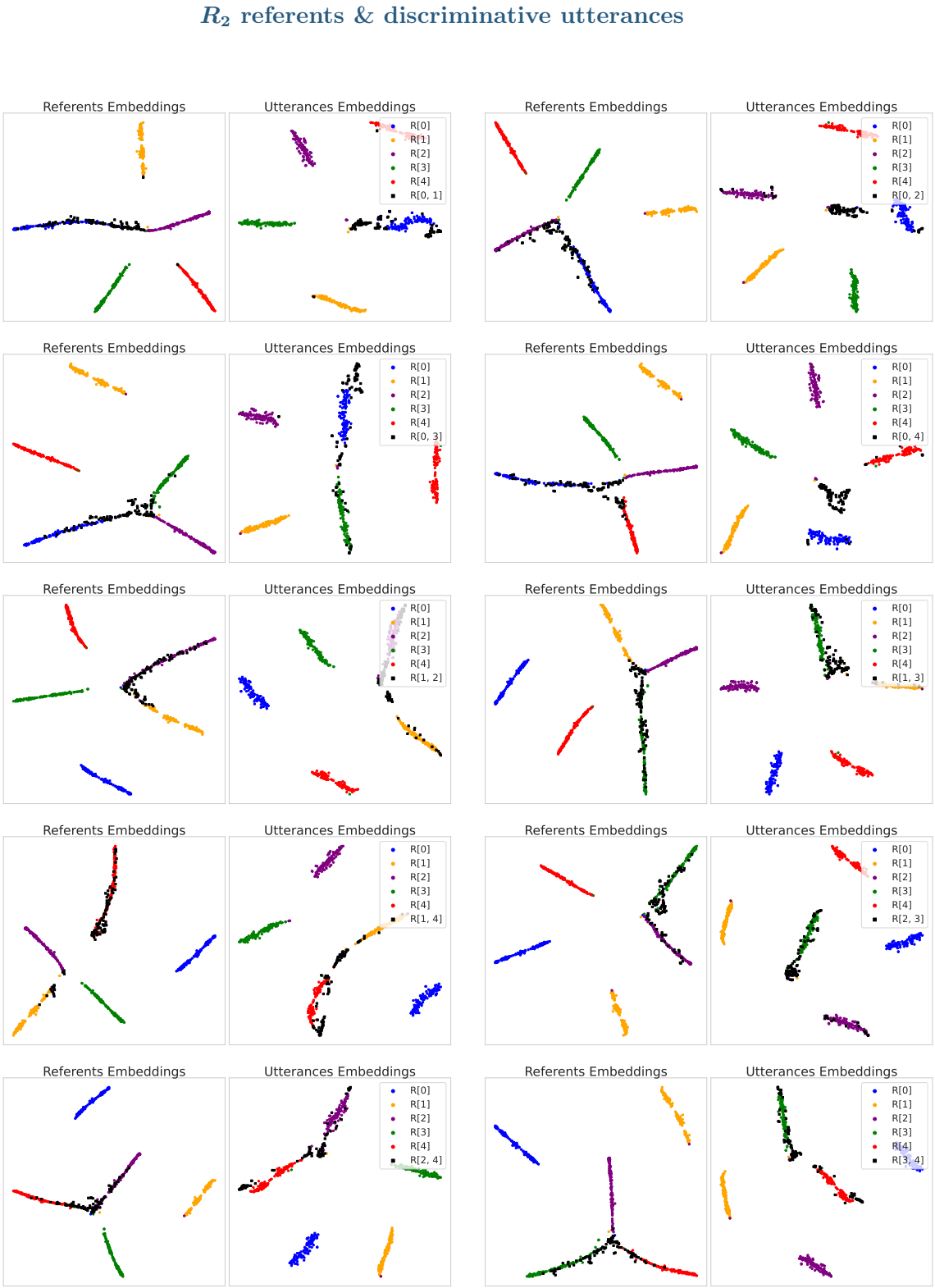


Figure B.15: **T-sne of referent and discriminative utterance embeddings.** Embeddings are computed for 100 perspectives of referents from R_2. Training conditions are unshared visual referents.

Appendix C

Grounding Spatio-Temporal Language with Transformers

C.1 Playground

Todo: *Merge with playground from imagine supplementary*

C.1.1 Environment

Temporal Playground is a procedurally generated environment consisting of 3 objects and an agent's body. There are 32 types of objects, listed in Fig. C.1 along with 5 object categories. Each object has a continuous 2D position, a size, a continuous color code specified by a 3D vector in RGB space, a type specified by a one-hot vector, and a boolean unit specifying whether it is grasped. Note that categories are not encoded in the objects' features. The agent's body has its 2D position in the environment and its gripper state (grasping or non-grasping) as features. The size of the body feature vector is 3 while the object feature vector has a size of 39. This environment is a spatio-temporal extension of the one used in this work Colas et al. (2020b).

All positions are constrained within $[-1, 1]^2$. The initial position of the agent is $(0, 0)$ while the initial object positions are randomized so that they are not in contact ($d(obj_1, obj_2) > 0.3$). Object sizes are sampled uniformly in $[0.2, 0.3]$, the size of the agent is 0.05. Objects can be grasped when the agent has nothing in hand, when it is close enough to the object center ($d(\text{agent}, obj) < (size(\text{agent}) + size(obj))/2$) and the gripper is closed (1, -1 when open). When a supply is on an animal or water is on a plant (contact define as distance between object being equal to the mean size of the two objects $d = (size(obj_1) + size(obj_2))/2$), the object will grow over time with a constant growth rate until it reaches the maximum size allowed for objects or until contact is lost.

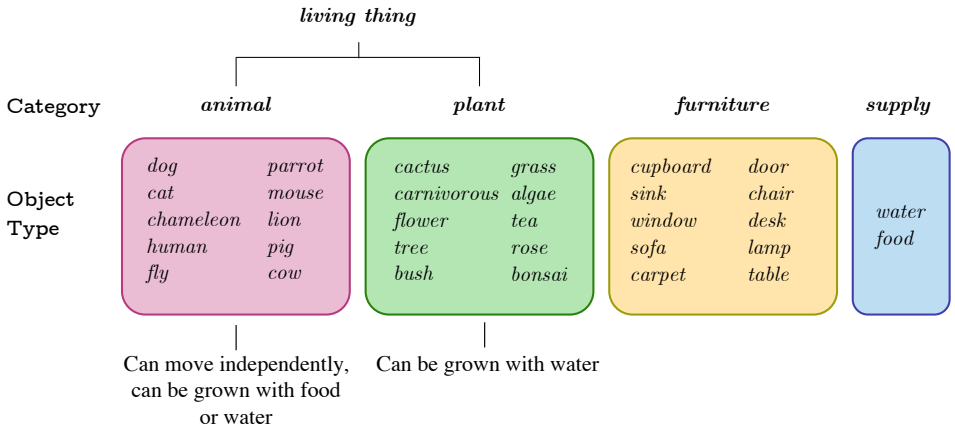


Figure C.1: **Representation of possible objects types and categories**. Information about the possible interactions between objects are also given.

C.1.2 Language

Grammar.

The synthetic language we use can be decomposed into two components: the instantaneous grammar and the temporal logic. Both are specified through the BNF given in Figure C.2.

Instantaneous grammar:

```
<S>             ::= <pred> <thing_A>
<pred>          ::= grow | grasp | shake
<thing_A>       ::= <thing_B> | <attr> <thing_B> | thing <localizer> |
                    thing <localizer_all>
<localizer>     ::= left of <thing_B> | right of <thing_B> |
                    bottom of <thing_B> | top of <thing_B>
<localizer_all> ::= left most | right most | bottom most | top most
<thing_B>       ::= dog | cat | ... | thing
<attr>          ::= blue | green | red
```

Temporal aspect:

```
<S>             ::= was <pred> <thing_A>
<thing_A>       ::= thing was <localizer> | thing was <localizer_all>
```

Figure C.2: **BNF of the grammar used in Temporal Playground**. The instantaneous grammar allows generating true sentences about predicates, spatial relations (one-to-one and one to all). These sentences are then processed by the temporal logic to produce the linguistic descriptions of our observations; this step is illustrated in the Temporal Aspect rules. See the main text for information on how these sentences are generated.

Concept Definition.

We split the set of all possible descriptions output by our grammar into four conceptual categories according to the rules given in Table C.1.

Concept	BNF	Size
1. Basic	<S> ::= <pred> <thing_A> <pred> ::= *grasp* <thing_A> ::= <thing_B> | <attr> <thing_B	152
2. Spatial	<S> ::= <pred> <thing_A> <pred> ::= *grasp* <thing_A> ::= <*thing* <localizer> | *thing* <localizer_all>	156
3. Temporal	<S> ::= <pred_A> <thing_A> | *was* <pred_B> <thing_A> <pred_A> ::= *grow* | *shake* <pred_B> ::= *grasp* | *grow* | *shake* <thing_A> ::= <thing_B> | <attr> <thing_B>	648
4. Spatio-Temporal	<S> ::= <pred_A> <thing_A> | *was* <pred_B> <thing_A> <pred_C> <thing_C> <pred_A> ::= *grow* | *shake* <pred_B> ::= *grasp* | *grow* | *shake* <pred_C> ::= *grasp* <thing_A> ::= *thing* <localizer> | *thing* <localizer_all> | *thing was* <localizer> | *thing was* <localizer_all> | <thing_C> ::= *thing was* <localizer> | *thing was* <localizer_all>	1716

Table C.1: **Concept categories with their associated BNF.** <thing_B>, <attr>, <localizer> and <localizer_all> are given in Fig. C.2

C.2 Supplementary Methods

C.2.1 Data Generation

Scripted bot.

To generate the traces matching the descriptions of our grammar we define a set of scenarii that correspond to sequences of actions required to fulfill the predicates of our grammar, namely *grasp*, *grow* and *shake*. Those scenarii are then conditioned on a boolean that modulates them to obtain a mix of predicates in the present and the past tenses. For instance, if a *grasp* scenario is sampled, there will be a 50% chance that the scenario will end with the object being grasped, leading to a present-tense description; and a 50% chance that the agent releases the object, yielding a past tense description.

Description generation from behavioral traces of the agent.

For each time step, the instantaneous grammar generates the set of all true instantaneous sentences using a set of filtering operations similar to the one used in CLEVR Johnson et al. (2016), without the past predicates and past spatial relations. Then the temporal logic component uses these linguistic traces in the following way: if a given sentence for a predicate is true in a past time step and false in the present time step, the prefix token *'was'* is prepended to the sentence; similarly, if a given spatial relation is observed in a previous time step and unobserved in the present, the prefix token *'was'* is prepended to the spatial relation.

C.2.2 Input Encoding

We present the input processing in Fig. C.3. At each time step t, the body feature vector b_t and the object features vector $o_{i,t}$, $i = 1, 2, 3$ are encoded using two single-layer neural networks whose output are of size h. Similarly, each of the words of the sentence describing the trace (represented as one-hot vectors) is encoded and projected in the dimension of size h. We concatenate to the vector obtained a modality token m that defines if the output belongs to the scene $(1, 0)$ or to the description $(0, 1)$. We then feed the resulting vectors to a positional encoding that modulates the vectors according to the time step in the trace for b_t and $o_{i,t}$, $i = 1, 2, 3$ and according to the position of the word in the description for w_l.

We call the encoded body features $\hat{b}_t$ and it corresponds to $\hat{S}_{0,t}$ of the input tensor of our model (see Fig. 7.2 in the Main document). Similarly, $\hat{o}_{i,t}$, $i = 1, 2, 3$ are the encoded object features corresponding to $\hat{S}_{i,t}$, $i = 1, 2, 3$. Finally $\hat{w}_l$ are the encoded words and the components of tensor $\hat{W}$.

We call h the hidden size of our models and recall that $|\hat{b}_t| = |\hat{o}_{i,t}| = |\hat{w}_l| = h + 2$. This parameter is varied during the hyper-parameter search.

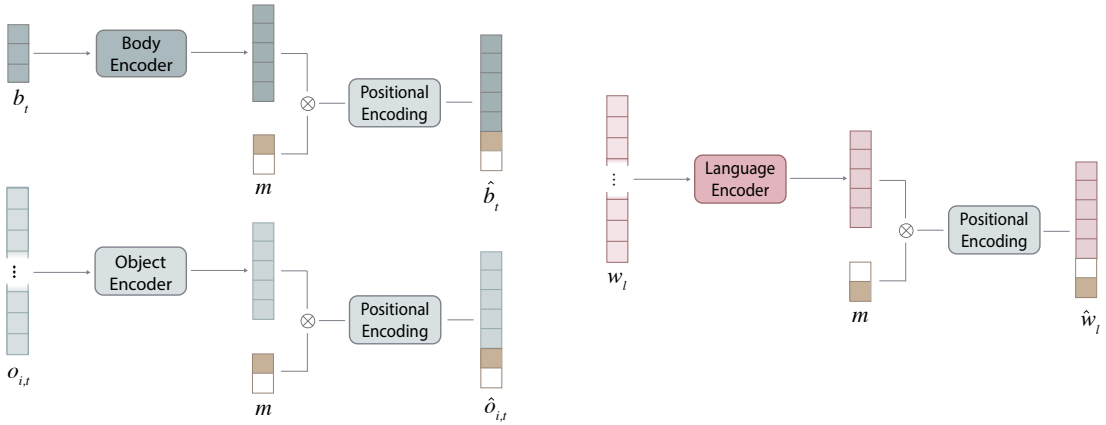


Figure C.3: **Diagram representing the projection of the inputs into the same dimension**

C.2.3 Details on LSTM models

To provide baseline models on our tasks we consider two LSTM variants. They are interesting baselines because they do not perform any relational computation except for relations between inputs at successive time steps. We consider the inputs as they were defined in Section **??** of the main paper. We consider two LSTM variants:

1. LSTM-FLAT: This variant has two internal LSTM: one that processes the language and one that processes the scenes as concatenations of all the body and object features. This produces two vectors that are concatenated into one, which is then run through an MLP and a final softmax to produce the final output.
2. LSTM-FACTORED: This variant independently processes the different body and object traces, which have previously been projected to the same dimension using a separate linear projection for the object and for the body. The language is processed by a separate LSTM. These body, object and language vectors are finally concatenated and fed to a final MLP and a softmax to produce the output.

C.2.4 Details on Training Schedule

Implementation Details.

The architectures are trained via backpropagation using the Adam OptimizerKingma & Ba (2017). The data is fed to the model in batches of 512 examples for 150 000 steps. We use a modular buffer to sample an important variety of different descriptions in each batch and to impose a ratio of positive samples of 0.1 for each description in each batch.

Model implementations.

We used the standard implementations of TransformerEncoderLayer and TransformerEncoder from pytorch version 1.7.1, as well as the default LSTM implementation. For initialization, we also use pytorch defaults.

Hyper-parameter search.

To pick the best set of parameters for each of our eight models, we train them on 18 conditions and select the best models. Note that each condition is run for 3 seeds and best models are selected according to their averaged F_1 score on randomly held-out descriptions (15% of the sentences in each category given in Table C.1).

Best models.

Best models obtained thanks to the parameter search are given in Table C.2.

Model	**Learning rate**	**Model hyperparams**			
		hidden size	layer count	head count	param count
UT	1e-4	256	4	8	1.3M
UT-WA	1e-5	512	4	8	14.0M
TFT	1e-4	256	4	4	3.5M
TFT-WA	1e-5	512	4	8	20.3M
SFT	1e-4	256	4	4	3.5M
SFT-WA	1e-4	256	2	8	2.7M
LSTM-FLAT	1e-4	512	4	N/A	15.6M
LSTM-FACTORED	1e-4	512	4	N/A	17.6M

Table C.2: Hyperparameters for all models

Robustness to hyperparameters

For some models, we have observed a lack of robustness to hyperparameters during our search. This translated to models learning to predict all observation-sentence tuples as false since the dataset is imbalanced (the proportion of true samples is 0.1). This behavior was systematically observed with a series of models whose hyperparameters are listed in Table C.3. This happens with the biggest models with high learning rates, especially with the -WA variants.

Model	Learning rate	Model hyperparams		
		hidden size	layer count	head count
UT-WA	1e-4	512	4	4
UT-WA	1e-4	512	4	8
SFT	1e-4	512	4	4
SFT-WA	1e-4	512	4	8
SFT-WA	1e-4	512	2	4
SFT-WA	1e-4	512	4	4
TFT	1e-4	512	4	4
TFT-WA	1e-4	512	4	8
TFT-WA	1e-4	512	2	4
TFT-WA	1e-4	512	4	4

Table C.3: Models and hyperparameters collapsing into uniform false prediction.

C.3 Supplementary Results

C.3.1 Generalization to new observations from known sentences

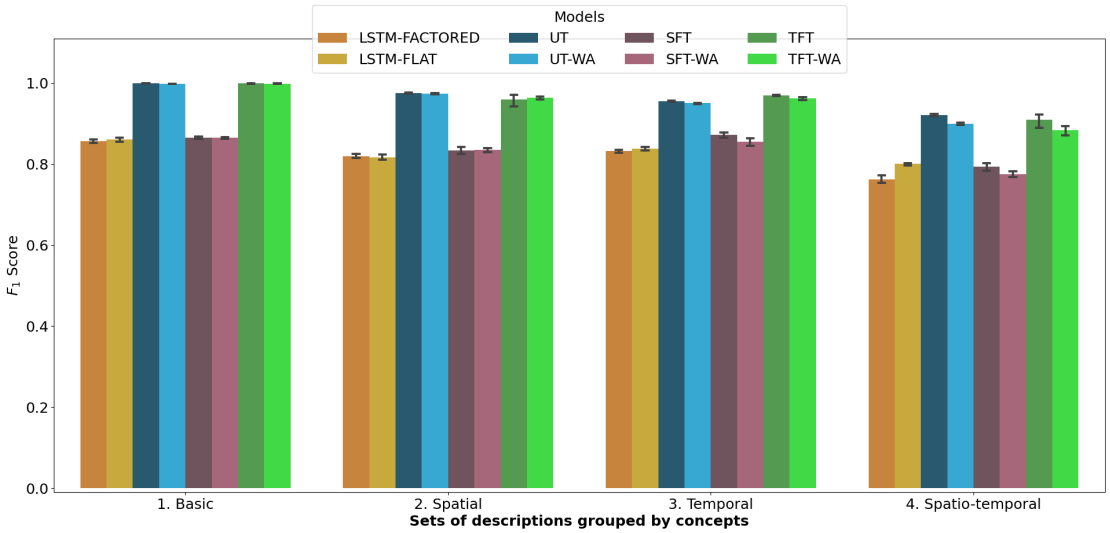


Figure C.4: F1 scores of all models on the train sentences with new observations.

In this section we shortly describe an additional evaluation setup we considered. We evaluate the model's f1-scores on sets of sentences that are seen as train sentences, but on newly generated observations. The results are plotted in Figure C.4.

C.3.2 Computing Resources

This work was performed using HPC resources from GENCI-IDRIS (Grant 2020-101594). We used 22k GPU-hours on nvidia-V100 GPUs for the development phase, hyperparameter search, and the main experiments.

Appendix D

Imagine

This supplementary material provides additional methods, results and discussion, as well as implementation details.

- Section D.1 gives a complete description of our setup and of the *Playground* environment.
- Section D.2 presents a focus on generalization and studies different types of generalization.
- Section D.3 presents a focus on exploration and how it is influenced by goal imagination.
- Section D.4 presents a focus on the goal imagination mechanism we use for IMAGINE.
- Section D.5 presents a focus on the *Modular-Attention* architecture.
- Section D.6 presents a focus on the benefits of learning the reward function.
- Section D.7 provides additional visualization of the goal embeddings and the attention vectors.
- Section D.8 discusses the comparison with goal-as-state approaches.
- Section D.9 gives all necessary implementation details.

D.1 Complete Description of the Playground Environment and Its Language

Environment description.

The environment is a 2D square: $[-1.2, 1.2]^2$. The agent is a disc of diameter 0.05 with an initial position $(0, 0)$. Objects have sizes uniformly sampled from $[0.2, 0.3]$ and their initial positions are randomized so that they are not in contact with each other. The agent has an action space of size 3 bounded in $[-1, 1]$. The first two actions control the agent's continuous 2D translation (bounded to 0.15 in any direction). The agent can grasp objects by getting in contact with them and closing its gripper (positive third action), unless it already has an object in hand. Objects include 10 animals, 10 plants, 10 pieces of furniture and 2 supplies. Admissible categories are *animal, plant, furniture, supply* and *living_thing* (animal or plant), see Figure D.1. Objects are assigned a color attribute (red, blue or green). Their precise color is a continuous RGB code uniformly

sampled from RGB subspaces associated with their attribute color. Each scene contains 3 of these procedurally-generated objects (see paragraph about the Social Partner below).

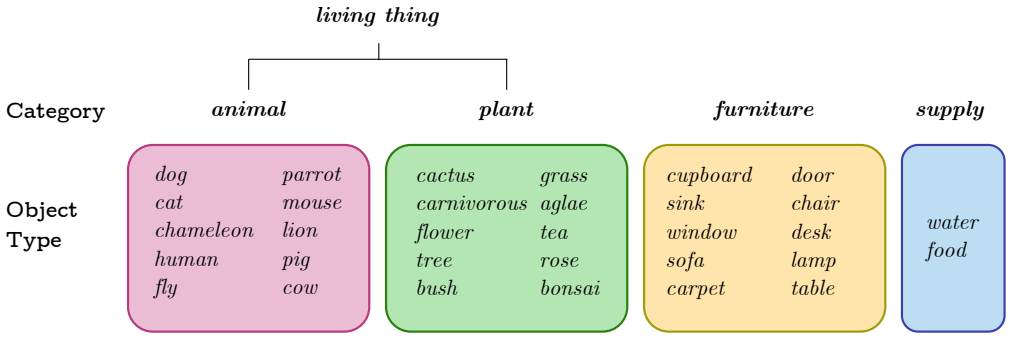


Figure D.1: Representation of possible objects types and categories.

Agent perception.

At time step t, we can define an observation $\mathbf{o}_t$ as the concatenation of body observations (2D-position, gripper state) and objects' features. These two types of features form affordances between the agent and the objects around. These affordances are necessary to understand the meaning of object interactions like *grasp*. The state $\mathbf{s}_t$ used as input of the models is the concatenation of $\mathbf{o}_t$ and $\mathbf{\Delta o}_t = \mathbf{o}_t - \mathbf{o}_0$ to provide a sense of time. This is required to acquire the understanding and behavior related to the *grow* predicate, as the agent needs to observe and produce a change in the object's size.

Social Partner.

SP has two roles:

- *Scene organization*: SP organize the scene according to the goal selected by the agent. When the agent selects a goal, it communicates it to SP. If the goal starts by the word *grow*, SP adds a procedurally-generated supply (water or food for animals, water for plants) of any size and color to the scene. If the goal contains an object (e.g. *red cat*), SP adds a corresponding object to the scene (with a procedurally generated size and RGB color). Remaining objects are generated procedurally. As a result, the objects required to fulfill a goal are always present and the scene contains between 1 (*grow* goals) and 3 (*go* goals) random objects. Note that all objects are procedurally generated (random initial position, RGB color and size).
- *Scene description*: SP provides NL descriptions of interesting outcomes experienced by the agent at the end of episodes. It takes the final state of an episode ($\mathbf{s}_T$) as input and returns matching NL descriptions: $\mathcal{D}_{\text{SP}}(\mathbf{s}_T) \subset \mathcal{D}^{\text{SP}}$. When SP provides *descriptions*, the agent considers them as targetable *goals*. This mapping $\mathcal{D}^{\text{SP}} \rightarrow \mathcal{G}^{\text{train}}$ simply consists in removing the first *you* token (e.g. turning *you grasp red door* into the goal *grasp red door*). Given the set of previously discovered goals

($\mathcal{G}_{\text{known}}$) and new descriptions $\mathcal{D}_{\text{SP}}(\mathbf{s}_T)$, the agent infers the set of goals that were not achieved: $\mathcal{G}_{\text{na}}(\mathbf{s}_T) = \mathcal{G}_{\text{known}} \setminus \mathcal{D}_{\text{SP}}(\mathbf{s}_T)$, where $\setminus$ indicates the complement.

Grammar.

We now present the grammar that generates descriptions for the set of goals achievable in the Playground environment ($\mathcal{G}^A$). **Bold** and { } refer to sets of words while *italics* refers to particular words:

1. Go: *(e.g. go bottom left)*
   - *go* + ***zone***
2. Grasp: *(e.g. grasp any animal)*
   - *grasp* + ***color*** $\cup$ {*any*} + ***object type*** $\cup$ ***object category***
   - *grasp* + *any* + ***color*** + *thing*
3. Grow: *(e.g. grow blue lion)*
   - *grow* + ***color*** $\cup$ {*any*} + ***living thing*** $\cup$ {*living_thing, animal, plant*}
   - *grow* + *any* + ***color*** + *thing*

   Word sets are defined by:

   - ***zone*** = {*center, top, bottom, right, left, top left, top right, bottom left, bottom right*}
   - ***color*** = {*red, blue, green*}
   - ***object type*** = ***living thing*** $\cup$ ***furniture*** $\cup$ ***supply***
   - ***object category*** = {*living_thing, animal, plant, furniture, supply*}
   - ***living thing*** = ***animal*** $\cup$ ***plant***
   - ***animal*** = {*dog, cat, chameleon, human, fly, parrot, mouse, lion, pig, cow*}
   - ***plant*** = {*cactus, carnivorous, flower, tree, bush, grass, algae, tea, rose, bonsai*}
   - ***furniture*** = {*door, chair, desk, lamp, table, cupboard, sink, window, sofa, carpet*}
   - ***supply*** = {*water, food*}
   - ***predicate*** = {*go, grasp, grow*}

We partition this set of achievable goals into a training ($\mathcal{G}^{\text{train}}$) and a testing ($\mathcal{G}^{\text{test}}$) set. Goals from $\mathcal{G}^{\text{test}}$ are intended to evaluate the ability of our agent to explore the set of achievable outcomes beyond the set of outcomes described by SP. The next section introduces this testing set and focuses on generalization. Note that some goals might be syntactically valid but not achievable. This includes all goals of the form *grow* + ***color*** $\cup$ {*any*} + ***furniture*** $\cup$ {*furniture*} (e.g. *grow red lamp*).

IMAGINE Pseudo-Code.

Algorithm 6 outlines the pseudo-code of our learning architecture. See Main Section 8.3.2 for high-level descriptions of each module and function.

Algorithm 6: IMAGINE

Input: env, SP
Initialize: L_e, $\mathcal{R}$, π, $mem(\mathcal{R})$, $mem(\pi)$, $\mathcal{G}_{\text{known}}$, $\mathcal{G}_{\text{im}}$
    # Random initializations for networks
    # empty sets for memories and goal sets
for $e = 1 : N_{episodes}$ **do**
  if $\mathcal{G}_{\text{known}} \neq \emptyset$ **then**
    sample g_{NL} from $\mathcal{G}_{\text{known}} \cup \mathcal{G}_{\text{im}}$
    $g \leftarrow L_e(g_{\text{NL}})$
  else
    sample g from $\mathcal{N}(0, \mathbf{I})$
  end if
  $s_0 \leftarrow$ env.reset()
  for $t = 1 : T$ **do**
    $a_t \leftarrow \pi(s_{t-1}, g)$
    $s_t \leftarrow$ env.step(a_t)
    mem_π.add(s_{t-1}, a_t, s_t)
  end for
  $\mathcal{G}_{\text{SP}} \leftarrow$ SP.get_descriptions(s_T)
  $\mathcal{G}_{\text{known}} \leftarrow \mathcal{G}_{\text{known}} \cup \mathcal{G}_{\text{SP}}$
  $mem(\mathcal{R})$.add(s_T, g_{NL}) for g_{NL} in $\mathcal{G}_{\text{SP}}$
  if goal imagination allowed **then**
    $\mathcal{G}_{\text{im}} \leftarrow$ **Imagination**$(\mathcal{G}_{\text{known}})$ # see Algorithm 7
  end if
  $\text{Batch}_\pi \leftarrow$ **ModularBatchGenerator**$(mem(\pi))$ # $\text{Batch}_\pi = \{(s, a, s')\}$
  $\text{Batch}_\pi \leftarrow$ **Hindsight**$(\text{Batch}_\pi, \mathcal{R}, \mathcal{G}_{\text{known}}, \mathcal{G}_{\text{im}})$ # $\text{Batch}_\pi = \{(s, a, r, g, s')\}$ where $r = \mathcal{R}(s, g)$
  $\pi \leftarrow$**RL_Update**(Batch_π)
  if e % reward_update_freq == 0 **then**
    $\text{Batch}_\mathcal{R} \leftarrow$ ModularBatchGenerator$(mem(\mathcal{R}))$
    $L_e, \mathcal{R} \leftarrow$ **LE&RewardFunctionUpdate**$(\text{Batch}_\mathcal{R})$
  end if
end for

D.2 Focus on Generalization

Because scenes are procedurally-generated, $\overline{\texttt{SR}}$ computed on $\mathcal{G}^{\text{train}}$ measures the generalization to new states. When computed on $\mathcal{G}^{\text{test}}$, however, $\overline{\texttt{SR}}$ measures both this state generalization and the generalization to new goal descriptions from $\mathcal{G}^{\text{test}}$. As $\overline{\texttt{SR}}_{\text{train}}$ is almost perfect, this section focuses solely on generalization in the language space: $\overline{\texttt{SR}}_{\text{test}}$.

Different types of generalization.

Generalization can occur in two different modules of the IMAGINE architecture: in the reward function and in the policy. Agents can only benefit from goal imagination when their reward function is able to generalize the meanings of imagined goals from the meanings of known ones. When they do, they can further train on imagined goals, which might, in turn, reinforce the generalization of the policy. This section characterizes different types of generalizations that the reward and policy can both demonstrate.

- Type 1 - *Attribute-object generalization*: This is the ability to accurately associate an attribute and an object that were never seen together before. To interpret the goal *grasp red tree* requires to isolate the *red* and *tree* concepts from other sentences and to combine them to recognize a *red tree*. To measure this ability, we removed from the training set all goals containing the following attribute-object combinations: *{blue door, red tree, green dog}* and added them to the testing set (4 goals).
- Type 2 - *Object identification*: This is the ability to identify a new object from its attribute. We left out of the training set all goals containing the word *flower* (4 goals). To interpret the goal *grasp red flower* requires to isolate the concept of *red* and to transpose it to the unknown object *flower*. Note that in the case of *grasp any flower*, the agent cannot rely on the attribute, and must perform some kind of complement reasoning:"if these are known objects, and that is unknown, then if must be a *flower*".
- Type 3 - *Predicate-category generalization*: This is the ability to interpret a predicate for a category when they were never seen together before. As explained in Section D.1, a category regroups a set of objects and is not encoded in the object state vector. It is only a linguistic concept. We left out all goals with the *grasp* predicate and the *animal* category (4 goals). To correctly interpret *grasp any animal* requires to identify objects that belong to the animal category (acquired from "growing *animal*" and "growing animal objects" goals), to isolate the concept of *grasping* (acquired from grasping non-*animal* objects) and to combine the two.
- Type 4 - *Predicate-object generalization*: This is the ability to interpret a predicate for an object when they were never seen together before. We leave out all goals with the *grasp* predicate and the *fly* object (4 goals). To correctly interpret *grasp any fly*, the agent should leverage its knowledge about the *grasp* predicate (acquired from the "grasping non-fly objects" goals) and the *fly* object (acquired from the "growing flies" goals).

- Type 5 - *Predicate dynamics generalization*: This is the ability to generalize the behavior associated with a predicate to another category of objects, for which the dynamics is changed. In the *Playground* environment, the dynamics of *grow* with ***animals*** and ***plants*** is a a bit different. ***animals*** can be grown with *food* and *water* whereas ***plants*** only grow with *water*. We want to see if IMAGINE can learn the dynamics of *grow* on ***animals*** and generalize it to ***plants***. We left out all goals with the *grow* predicate and any of the ***plant*** objects, *plant* and *living thing* categories (48 goals). To interpret, *grow any plant*, the agent should be able to identify the ***plant*** objects (acquired from the "grasping plants" goals) and that objects need supplies (food or water) to *grow* (acquired from the "growing animals" goals). Type 5 is more complex than Type 4 for two reasons: 1) because the dynamics change and 2) because it mixes objects and categories. Note that, by definition, the zero-shot generalization is tested without additional reward signals (before imagination). As a result, even the best zero-shot generalization possible cannot adapt the *grow* behavior from animals to plant and would bring food and water with equal probability $p = 0.5$ for each.

Table D.1 provides the exhaustive list of goals used to test each type of generalization.

Different ways to generalize.

Agent can generalize to out-of-distribution goals (from any of the 5 categories above) in three different ways:

1. *Policy zero-shot generalization*: The policy can achieve the new goal without any supplementary training.

2. *Reward zero-shot generalization*: The reward can tell whether the goal is achieved or not without any supplementary training.

3. *Policy n-shot generalization or behavioral adaptation*: When allowed to imagine goals, IMAGINE agents can use the zero-shot generalization of their reward function to autonomously train their policy to improve on imagined goals. After such training, the policy might show improved generalization performance compared to its zero-shot abilities. We call this performance *n-shot generalization*. The policy received supplementary training, but did not leverage any external supervision, only the zero-shot generalization of its internal reward function. This is crucial to achieve Type 5 generalization. As we said, zero-shot generalization cannot figure out that plants only grow with water. Fine-tuning the policy based on experience and internal rewards enables agents to perform *behavioral adaptation*: adapting their behavior with respect to imagined goals in an autonomous manner (see Main Figure 8.3).

Table D.1: Testing goals in $\mathcal{G}^{\text{test}}$, by type.

Type 1	*Grasp blue door*, *Grasp green dog*,*Grasp red tree*, *Grow green dog*
Type 2	*Grasp any flower*, *Grasp blue flower*, *Grasp green flower*, *Grasp red flower*, *Grow any flower*, *Grow blue flower*, *Grow green flower*, *Grow red flower*
Type 3	*Grasp any animal*, *Grasp blue animal*, *Grasp green animal*, *Grasp red animal*
Type 4	*Grasp any fly*, *Grasp blue fly*, *Grasp green fly*, *Grasp red fly*
Type 5	*Grow any algae*, *Grow any bonsai*, *Grow any bush*, *Grow any cactus* *Grow any carnivorous*, *Grow any grass*, *Grow any living_thing*, *Grow any plant* *Grow any rose*, *Grow any tea*, *Grow any tree*, *Grow blue algae* *Grow blue bonsai*, *Grow blue bush*,*Grow blue cactus*, *Grow blue carnivorous* *Grow blue grass*, *Grow blue living_thing*, *Grow blue plant*, *Grow blue rose* *Grow blue tea*, *Grow blue tree*,*Grow green algae*, *Grow green bonsai* *Grow green bush*, *Grow green cactus*, *Grow green carnivorous*, *Grow green grass* *Grow green living_thing*, *Grow green plant*, *Grow green rose*, *Grow green tea* *Grow green tree*, *Grow red algae*, *Grow red bonsai*, *Grow red bush* *Grow red cactus*, *Grow red carnivorous*, *Grow red grass*, *Grow red living_thing* *Grow red plant*, *Grow red rose*, *Grow red tea*, *Grow red tree*

Experiments.

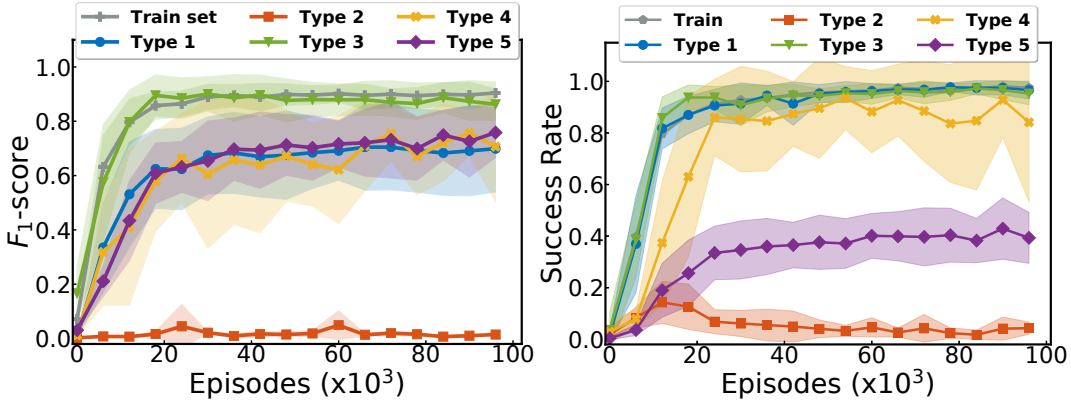
Figure D.2 presents training and generalization performance of the reward function and policy. We evaluate the generalization of the reward function via its average F_1 score on $\mathcal{G}^{\text{test}}$, the generalization of the policy by $\overline{\texttt{SR}}_{\text{test}}$.

Reward function zero-shot generalization. When the reward function is trained in parallel of the policy, we monitor its zero-shot generalization capabilities by computing the F_1-score over a dataset collected separately with a trained policy run on goals from $\mathcal{G}^{\text{test}}$ (kept fixed across runs for fair comparisons). As shown in Figure D.2a, the reward function exhibits good zero-shot generalization properties over 4 types of generalization after 25×10^3 episodes. Note that, because we test on data collected with a different RL policy, the F_1-scores presented in Figure D.2a may not faithfully describe the true generalization of the reward function during co-training.

Policy zero-shot generalization. The zero-shot performance of the policy is evaluated in Figure D.2b (*no imagination* condition) and in the period preceding goal imagination in Figure D.2c and D.2d (before vertical dashed line). The policy shows excellent zero-shot generalization properties for Type 1, 3 and 4, average zero-shot generalization on Type 5 and fails to generalize on Type 2. Type 1, 3 and 4 can be said to have similar levels of difficulty, as they all require to learn two concepts individually before combining them at test time. Type 2 is much more difficult as the meaning of only one word is known. The language encoder indeed receives a new word token which seems to disturb behavior. As said earlier, zero-shot generalization on Type 5 cannot do better than 0.5, as it cannot infer that plants only require water.

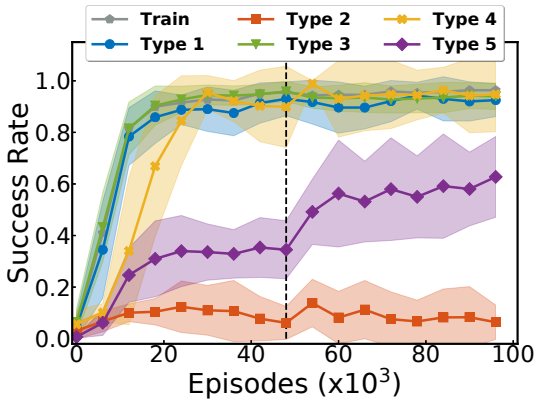
Policy n-shot generalization. When goal imagination begins (Figures D.2c and D.2d

after the vertical line), agents can imagine goals and train on them. This means that $\overline{\texttt{SR}}$ evaluates n-shot policy generalization. Agents can now perform *behavior adaptation*. They can learn that plants need water. As they learn this, their generalization performance on goals from Type 5 increases and goes beyond 0.5. Note that this effects fights the zero-shot generalization. By default, policy and reward function apply zero-shot generalization: e.g. they bring water or food equally to plants. Behavioral adaptation attempts to modify that default behavior. Because of the poor zero-shot generalization of the reward on goals of Type 2, agents cannot hope to learn Type 2 behaviors. Moreover, Type 2 goals cannot be imagined, as the word *flower* is unknown to the agent.

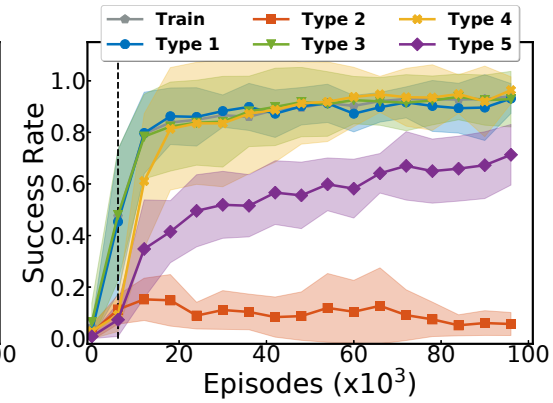


(a) **Reward Function, no imagination**

(b) **Policy, no imagination**



(c) **Policy, imagination half way**



(d) **Policy, imagination early**

Figure D.2: **Zero-shot and n-shot generalizations of the reward function and policy.** Each figure represents the training and testing performances (split by generalization type) for the reward (a), and the policy (b, c, d). (a) and (b) represent zero-shot performance in the *no imagination* conditions. In (c) and (d), agents start to imagine goals as denoted by the vertical dashed line. Before that line, $\overline{\texttt{SR}}$ evaluate zero-shot generalization. After, it evaluates the n-shot generalization, as agent can train autonomously on imagined goals.

D.3 Focus on exploration

Interesting Interactions.

Interesting interactions are trajectories of the agent that humans could infer as goal-directed. If an agent brings water to a plant and grows it, it makes sense for a human. If it then tries to do this for a lamp, it also feels goal-directed, even though it does not work. This type of behavior characterizes the penchant of agents to interact with objects around them, to try new things and, as a result, is a good measure of exploration.

Sets of interesting interactions.

We consider three sets of interactions: 1) interactions related to training goals; 2) to testing goals; 3) the extra set. This *extra set* contains interactions where the agent brings water or food to a piece of furniture or to another supply. Although such behaviors do not achieve any of the goals, we consider them as interesting exploratory behaviors. Indeed, they testify that agents try to achieve imagined goals that are meaningful from the point of view of an agent that does not already know that doors cannot be grown, i.e. corresponding to a meaningful form of generalization after discovering that animals or plants can be grown (e.g. *grow any door*).

The Interesting Interaction Count metric.

We count the number of interesting interactions computed over all final transitions from the last 600 episodes (1 epoch). Agents do not need to target these interactions, we just report the number of times they are experienced. Indeed, the agent does not have to target a particular interaction for the trajectory to be interesting from an exploratory point of view. The HER mechanism ensures that these trajectories can be replayed to learn about any goal, imagined or not. Computed on the extra set, the *Interesting Interaction Count* (I2C) is the number of times the agent was found to bring supplies to a furniture or to other supplies over the last epoch:

$$\text{I2C}_{\text{extra}} = \sum_{i \in \mathcal{I}=\mathcal{G}_{\text{extra}}} \sum_{t=1}^{600} \delta_{i,t},$$

where $\delta_{i,t} = 1$ if interaction i was achieved in episode t, 0 otherwise and $\mathcal{I}$ is the set of interesting interactions (here from the extra set) performed during an epoch.

Agents that are allowed to imagine goals achieve higher scores in the testing and extra sets of interactions, while maintaining similar exploration scores on the training set, see Figures D.3a to D.3c.

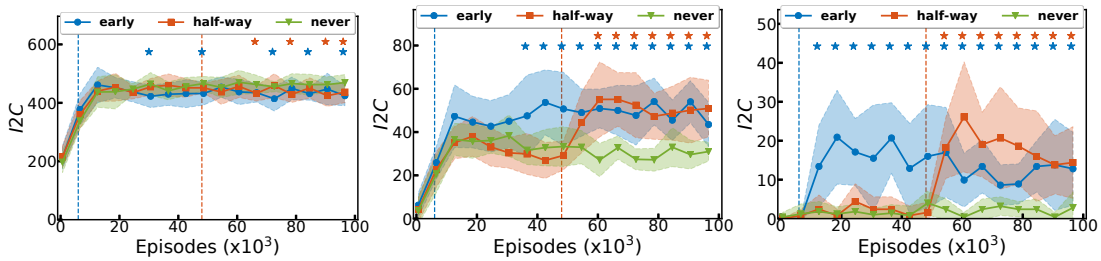


Figure D.3: **Exploration metrics** (a) Interesting interaction count (I2C) on training set, (b) I2C on testing set, (c) I2C on extra set. Goal imagination starts early (vertical blue line), half-way (vertical orange line) or does not start (*no imagination* baseline in green).

D.4 Focus on Goal Imagination

Algorithm 7 presents the algorithm underlying our goal imagination mechanism. This mechanism is inspired from the *Construction Grammar* (CG) literature and generates new sentences by composing known ones Goldberg (2003). It computes sets of equivalent words by searching for sentences with an edit distance of 1: sentences where only one word differs. These words are then labelled equivalent, and can be substituted in known sentences. Note that the goal imagination process filters goals that are already known. Although all sentences from $\mathcal{G}^{\text{train}}$ can be imagined, there are filtered out of the imagined goals as they are discovered. Imagining goals from $\mathcal{G}^{\text{train}}$ before they are discovered drives the exploration of IMAGINE agents. In our setup, however, this effect remains marginal as all the goals from $\mathcal{G}^{\text{train}}$ are discovered in the first epochs (see Figure D.5).

Algorithm 7: Goal Imagination. The edit distance between two sentences refers to the number of words to modify to transform one sentence into the other.

Input: $\mathcal{G}_{\text{known}}$ (discovered goals)
Initialize: *word_eq* (list of sets of equivalent words, empty)
Initialize: *goal_template* (list of template sentences used for imagining goals, empty)
Initialize: $\mathcal{G}_{\text{im}}$ (empty)
for g_{NL} in $\mathcal{G}_{\text{known}}$ **do** {Computing word equivalences}
  new_goal_template = True
  for g_m in *goal_template* **do**
    if edit_distance(g_{NL}, g_m) < 2 **then**
      new_goal_template = False
      if edit_distance(g_{NL}, g_m) == 1 **then**
        $w_1, w_2 \leftarrow$ *get_non_matching_words*(g_{NL}, g_m)
        if w_1 and w_2 not in any of *word_eq* sets **then**
          word_eq.add($\{w_1, w_2\}$)
        else
          for *eq_set* in *word_eq* **do**
            if $w_1 \in$ *eq_set* or $w_2 \in$ *eq_set* **then**
              eq_set = *eq_set* $\cup \{w_1, w_2\}$
            end if
          end for
        end if
      end if
    end if
  end for
  if *new_goal_template* **then**
    goal_template.add(g_{NL})
  end if
end for
for g in *goal_template* **do** {Generating new sentences}
  for w in g **do**
    for *eq_set* in *word_eq* **do**
      if $w \in$ *eq_set* **then**
        for w' in *eq_set* **do**
          $g_{im} \leftarrow$ replace(g, w, w')
          if $g_{im} \notin \mathcal{G}_{\text{known}}$ **then**
            $\mathcal{G}_{\text{im}} = \mathcal{G}_{\text{im}} \cup \{g_{im}\}$
          end if
        end for
      end if
    end for

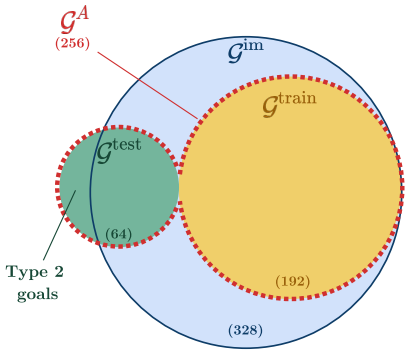


Figure D.4: Venn diagram of goal spaces.

Table D.2: All imaginable goals $\mathcal{G}^{\text{im}}$ generated by the Construction Grammar Heuristic.

Goals from $\mathcal{G}^{\text{train}}$	$\mathcal{G}^{\text{train}}$. (Note that known goals are filtered from the set of imagined goals. However, any goal from $\mathcal{G}^{\text{train}}$ can be imagined before it is encountered in the interaction with SP.)
Goals from $\mathcal{G}^{\text{test}}$	All goals from Type 1, 3, 4 and 5, see Table D.1
Syntactically incorrect goals	*Go bottom top*, *Go left right*, *Grasp red blue thing*, *Grow blue red thing*, *Go right left*, *Go top bottom*, *Grasp green blue thing*, *Grow green red thing*, *Grasp green red thing* *Grasp blue green thing*, *Grasp blue red thing*, *Grasp red green thing*.
Syntactically correct but unachievable goals	*Go center bottom*, *Go center top*, *Go right center*, *Go right bottom*, *Go right top*, *Go left center*, *Go left bottom*, *Go left top*, *Grow green cupboard*, *Grow green sink*, *Grow blue lamp*, *Go center right*, *Grow green window*, *Grow blue carpet*, *Grow red supply*, *Grow any sofa*, *Grow red sink*, *Grow any chair*, *Go top center*, *Grow blue table*, *Grow any door*, *Grow any lamp*, *Grow blue sink*, *Go bottom center*, *Grow blue door*, *Grow blue supply*, *Grow green carpet*, *Grow blue furniture*, *Grow green supply*, *Grow any window*, *Grow any carpet*, *Grow green furniture*, *Grow green chair*, *Grow green food*, *Grow any cupboard*, *Grow red food*, *Grow any table*, *Grow red lamp* , *Grow red door*, *Grow any food*, *Grow blue window*, *Grow green sofa*, *Grow blue sofa*, *Grow blue desk*, *Grow any sink*, *Grow red cupboard*, *Grow green door*, *Grow red furniture*, *Grow blue food*, *Grow red desk* , *Grow red table*, *Grow blue chair*, *Grow red sofa*, *Grow any furniture*, *Grow red window*, *Grow any desk*, *Grow blue cupboard*, *Grow red chair*, *Grow green desk*, *Grow green table*, *Grow red carpet*, *Go center left*, *Grow any supply*, *Grow green lamp*, *Grow blue water*, *Grow red water*, *Grow any water*, Grow green water, *Grow any water*, Grow green water.

Imagined goals.

We run our goal imagination mechanism based on the Construction Grammar Heuristic (CGH) from $\mathcal{G}^{\text{train}}$. After filtering goals from $\mathcal{G}^{\text{train}}$, this produces 136 new imagined sentences. Table D.2 presents the list of these goals while Figure D.4 presents a Venn diagram of the various goal sets. Among these 136 goals, 56 belong to the testing set $\mathcal{G}^{\text{test}}$. This results in a coverage of 87.5% of $\mathcal{G}^{\text{test}}$, and a precision of 45%. In goals that do not belong to $\mathcal{G}^{\text{test}}$, goals of the form *Grow* + $\{any\} \cup$ **color** + **furniture** $\cup$ **supplies** (e.g. *Grow any lamp*) are *meaningful* to humans, but are not achievable in the environment (*impossible*).

Variants of goal imagination mechanisms.

Main Section 8.4.2 investigates variants of our goal imagination mechanisms:

1. *Lower coverage:* To reduce the coverage of CGH while maintaining the same precision, we simply filter half of the goals that would have been imagined by CGH. This filtering is probabilistic, resulting in different imagined sets for different runs. It happens online, meaning that the coverage is always half of the coverage that CGH would have had at the same time of training.

2. *Lower precision:* To reduce precision while maintaining the same coverage, we sample a random sentence (random words from the words of $\mathcal{G}^{\text{train}}$) for each goal imagined by CGH that does not belong to $\mathcal{G}^{\text{test}}$. Goals from $\mathcal{G}^{\text{test}}$ are still imagined via the CGH mechanism. This variants only doubles the imagination of sentences that do not belong to $\mathcal{G}^{\text{test}}$.

3. *Oracle:* Perfect precision and coverage is achieved by filtering the output of CGH, keeping only goals from $\mathcal{G}^{\text{test}}$. Once the 56 goals that CGH can imagine are imagined, the oracle variants adds the 8 remaining goals: those including the word *flower* (Type 2 generalization).

4. *Random goals:* Each time CGH would have imagined a new goal, it is replaced by a randomly generated sentence, using words from the words of $\mathcal{G}^{\text{train}}$.

Note that all variants imagine goals at the same speed as the CGH algorithm. They simply filter or add noise to its output, see Figure D.5.

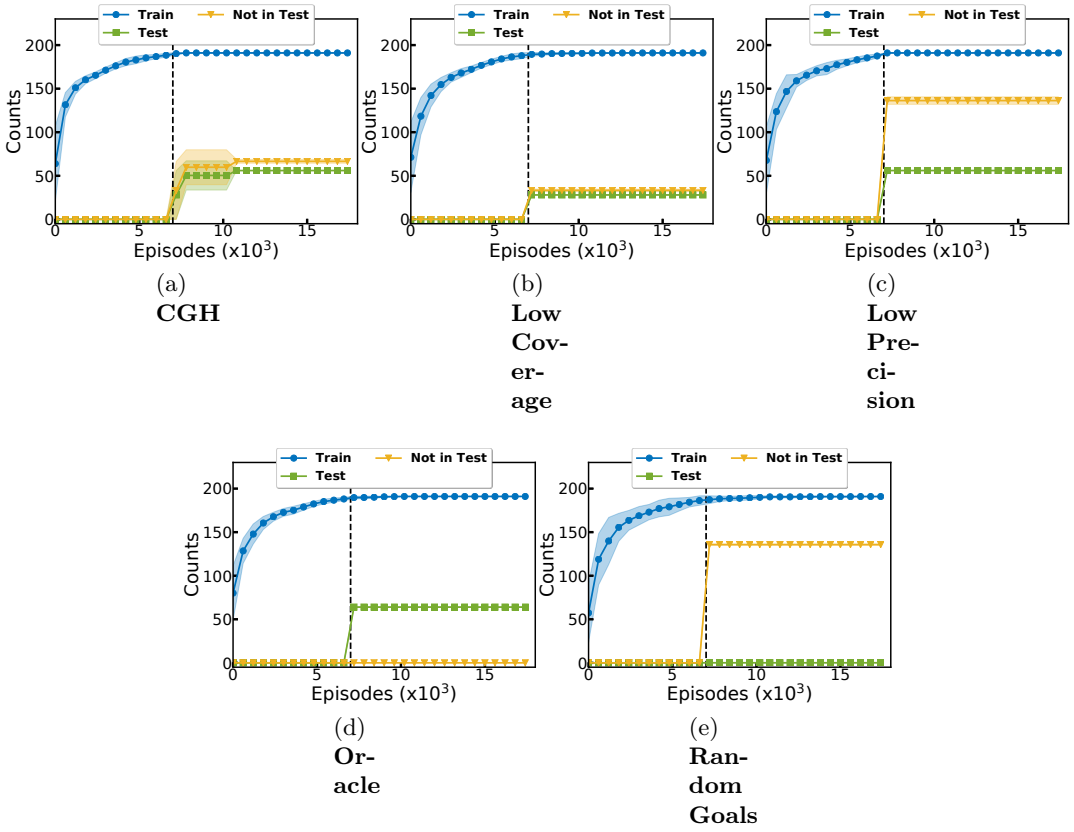


Figure D.5: **Evolution of known goals for various goal imagination mechanisms.** All graphs show the evolution of the number of goals from $\mathcal{G}^{\text{train}}$, $\mathcal{G}^{\text{test}}$ and others in the list of known goals $\mathcal{G}_{\text{known}}$. We zoom on the first epochs, as most goals are discovered and invented early. Vertical dashed line indicates the onset of goal imagination. (a) CGH; (b) Low Coverage; (c) Low precision; (d) Oracle; (e) Random Goals.

Effect of low coverage on generalization.

In Main Section 8.4.2, we compare our goal imagination mechanism to a *Low Coverage* variant that only covers half of the proportion of $\mathcal{G}^{\text{test}}$ covered by CGH (44%). Figure D.6 shows that the generalization performance on goals from $\mathcal{G}^{\text{test}}$ that the agent imagined (n-shot generalization, blue) are not significantly higher than the generalization performance on goals from $\mathcal{G}^{\text{test}}$ that were not imagined (zero-shot generalization). As they are both significantly higher than the *no imagination* baseline, this implies that training on imagined goals boosts zero-shot generalization on similar goals that were not imagined.

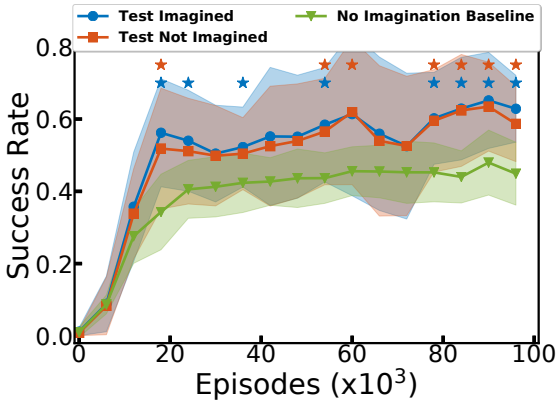


Figure D.6: **Zero-shot versus n-shot.** We look at the *Low Coverage* variant of our goal imagination mechanism that only covers 43.7% the test set with a 45% precision. We report success rates on testing goals of Type 5 (*grow + plant*) and compare with the *no imagination* baseline (green). We split in two: goals that were imagined (blue), and goals that were not (orange).

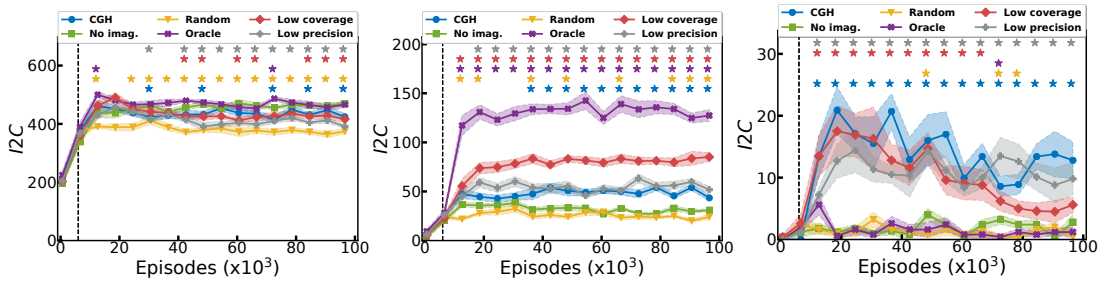
Details on the impacts of various goal imagination mechanisms on exploration.

Figure D.7 presents the I2C exploration scores on the training, testing and extra sets for the different goal imagination mechanisms introduced in Main Section 8.4.2. Let us discuss each of these scores:

1. *Training interactions.* In Figure D.7a, we see that decreasing the precision (Low Precision and Random Goal conditions) affects exploration on interactions from the training set, where it falls below the exploration of the *no imagination* baseline. This is due to the addition of meaningless goals forcing agent to allow less time to meaningful interactions relatively.

2. *Testing interactions.* In Figure D.7b, we see that the highest exploration scores on interactions from the test set comes from the oracle. Because it shows high coverage and precision, its spends more time on the diversity of interactions from the testing set. What is more surprising is the exploration score of the low coverage condition, higher than the exploration score of CGH. With an equal precision, CGH should show better exploration, as it covers more test goals. However, the *Low Coverage* condition, by spending more time exploring each of its imagined goals (it imagined fewer), probably learned to master them better, increasing the robustness of its behavior towards those. This insight advocates for the use of goal selection methods based on learning progress Forestier & Oudeyer (2016); Colas et al. (2019a). Agents could estimate their learning progress on imagined goals using their internal reward function and its zero-shot generalization. Focusing on goals associated to high learning progress might help agents filter goals they can learn about from others.

3. *Extra interactions.* Figure D.7c shows that only the goal imagination mechanisms that invent goals not covered by the testing set manage to boost exploration in this extra set. The oracle perfectly covers the testing set, but does not generate goals related to other objects (e.g. *grow any lamp*).



(a) **I2C on $\mathcal{G}^{\text{train}}$** (b) **I2C on $\mathcal{G}^{\text{test}}$** (c) **I2C on $\mathcal{G}^{\text{extra}}$**

Figure D.7: **Exploration metrics for different goal imagination mechanisms**: (a) Interesting interaction count (I2C) on training set, I2C on testing set, (c) I2C on extra set. Goal imagination starts early (vertical line), except for the *no imagination* baseline (green). Standard errors of the mean plotted for clarity (as usual, 10 seeds).

D.5 Focus on Architectures

This section compares our proposed object-based modular architecture MA for the policy and reward function to a flat architecture that does not use inductive biases for efficient skill transfer. We hypothesize that only the object-based modular architectures enable a generalization performance that is sufficient for the goal imagination to have an impact on generalization and exploration. Indeed, when generalization abilities are low, agents cannot evaluate their performance on imagined goals and thus, cannot improve.

Preliminary study of the reward function architecture.

We first compared the use of modular and flat architectures for the reward function (MA$^{\mathcal{R}}$ vs FA$^{\mathcal{R}}$ in Figure D.8). This experiment was conducted independently from policy learning, in a supervised setting. We use a dataset of 50×10^3 trajectories and associated goal descriptions collected using a pre-trained policy. To closely match the training conditions of IMAGINE, we train the reward function on the final states s_T and test it on any states s_t, $t = [1, .., T]$ of other episodes. Table D.3 provides the F_1 score computed at convergence on $\mathcal{G}^{\text{train}}$ and $\mathcal{G}^{\text{test}}$ for the two architectures.

Table D.3: Reward function architectures performance.

	$F_{1\text{train}}$	$F_{1\text{test}}$
MA$^{\mathcal{R}}$	0.98 ± 0.02	0.64 ± 0.22
FA$^{\mathcal{R}}$	0.60 ± 0.10	0.22 ± 0.05

MA$^{\mathcal{R}}$ outperforms FA$^{\mathcal{R}}$ on both the training and testing sets. In addition to its poor generalization performance, FA$^{\mathcal{R}}$'s performance on the training set are too low to support policy learning. As a result, the remaining experiments in this paper use the MA$^{\mathcal{R}}$ architecture for all reward functions. Thereafter, MA is always used for the reward function and the terms MA and FA refer to the architecture of the policy.

Architectures representations.

The combination of MA for the reward function and either MA or FA for the policy are represented in Figure D.9.

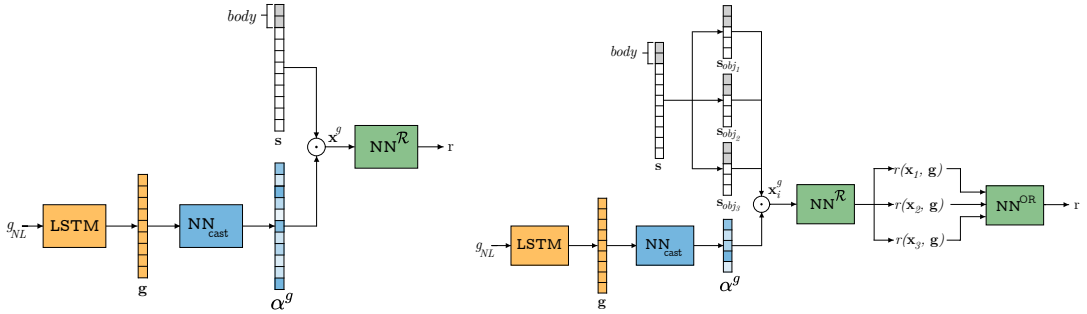
Policy architecture comparison.

Table D.4 shows that MA significantly outperforms FA on both the training and testing sets at convergence. Figure D.10a clearly shows an important gap between the generalization performance of the modular and the flat architecture. In average, less than 20% of the testing goals can be achieved with FA when MA masters half of them without imagination. Moreover, there is no significant difference between the never and the early

imagination conditions for the flat architecture. The generalization boost enabled by the imagination is only observable for the modular architecture (see Main Table 8.1). Figure D.10c and D.10d support similar conclusions for exploration: only the modular architecture enable goal imagination to drive an exploration boost on the testing and extra sets of interactions.

Table D.4: Architectures performance. Both p-values $< 10^{-10}$.

	$\overline{\text{SR}}_{\text{train}}$	$\overline{\text{SR}}_{\text{test}}$
MA	0.95 ± 0.05	0.76 ± 0.10
FA	0.40 ± 0.13	0.16 ± 0.06



(a) FA$^{\mathcal{R}}$ (b) MA$^{\mathcal{R}}$

Figure D.8: **Reward function architectures**: (a) *Flat-attention* reward function (FA$^{\mathcal{R}}$) and (b) *Modular-attention* reward function (MA$^{\mathcal{R}}$). We use MA$^{\mathcal{R}}$ for all experiments except for the experiment in Table D.3

In preliminary experiments, we tested a *Flat-Concatenation* (FC) architecture where the gated attention mechanism was replaced by a simple concatenation of goal encoding to the state vector. We did not found signficant difference with respect to FA. We chose to pursue with the attention mechanism, as it improves model interpretability (see Additional Visualization D.7).

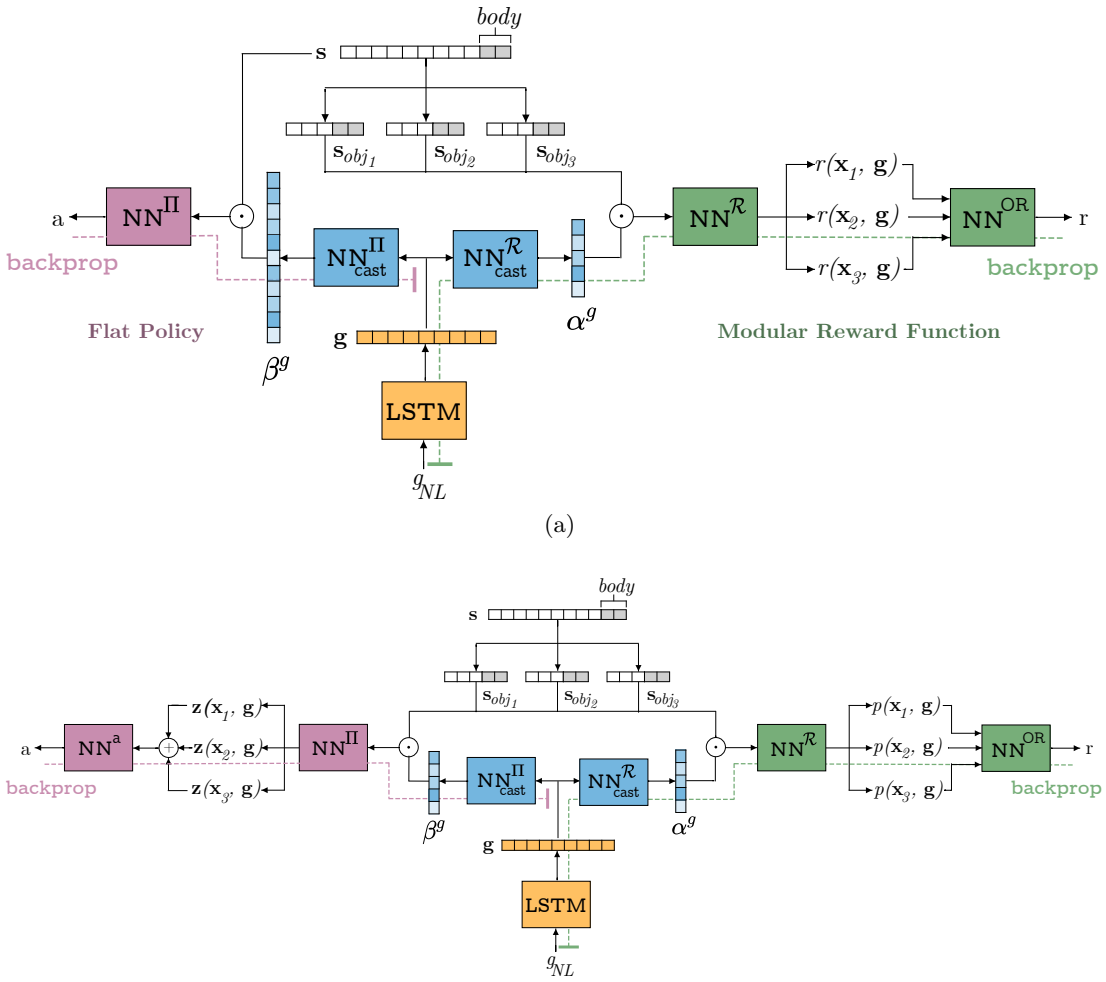
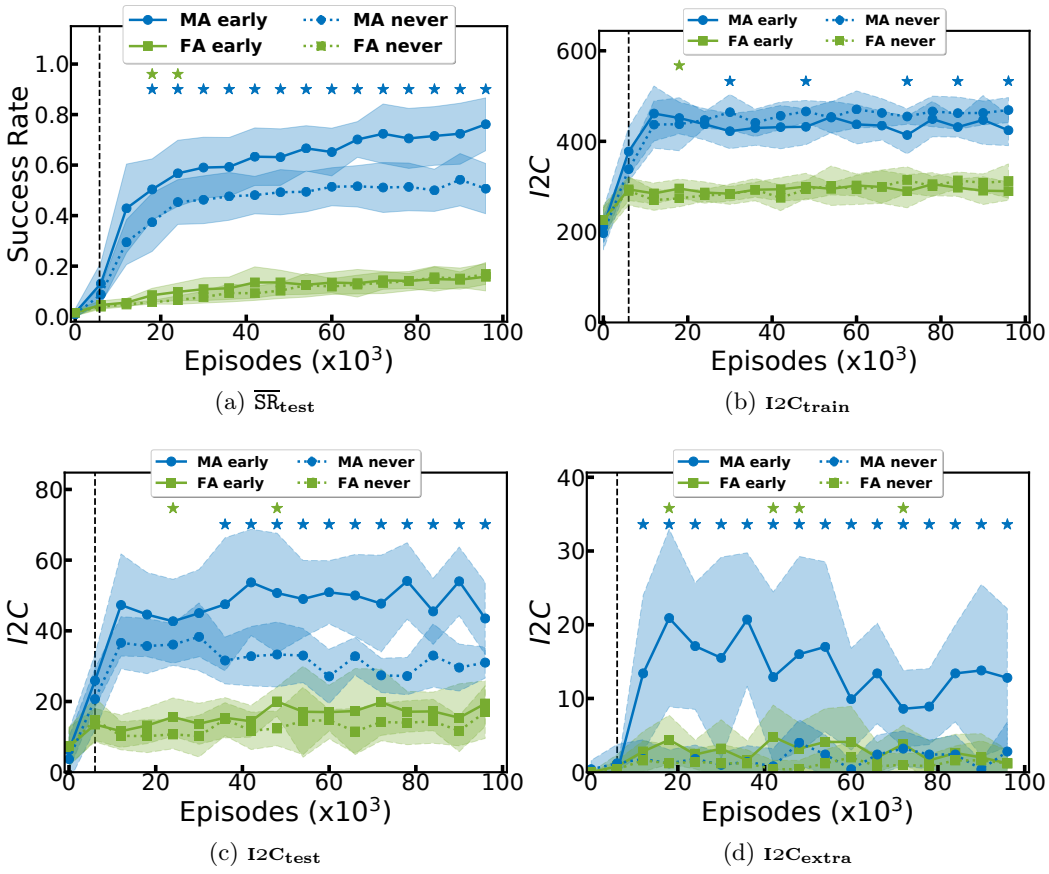


Figure D.9: **Policy and reward function architectures:** (a) *Modular-attention* (MA) reward + *Flat-attention* (FA) policy. (b) MA reward + MA policy. In both figures, the reward function is represented on the right in green, the policy on the left in pink, the language encoder in the bottom in yellow and the attention mechanisms at the center in blue.



(a) $\overline{\text{SR}}_{\text{test}}$ (b) $\textsc{i2c}_{\textbf{train}}$

(c) $\textsc{i2c}_{\textbf{test}}$ (d) $\textsc{i2c}_{\textbf{extra}}$

Figure D.10: **Policy architecture comparison:** (a) $\overline{\text{SR}}$ on $\mathcal{G}^{\text{test}}$ for the FA and MA architectures when the agent starts imagining goals early (plain, after the black vertical dashed line) or never (dashed). (b, c, d) I2C on interactions from the training, testing and extra sets respectively. Imagination is performed using CGH. Stars indicate significant differences between CGH and the corresponding *no imagination* baseline.

D.6 Focus on Reward Function

Our IMAGINE agent is autonomous and, as such, needs to learn its own reward function. It does so by leveraging a weak supervision from a social partner that provides descriptions in a simplified language. This reward function can be used for many purposes in the architecture. This paper leverages some of these ideas (the first two), while others are left for future work (the last two):

- **Behavior Adaptation.** As Main Section 8.4.1 showed, the reward function enables agents to adapt their behavior with respect to imagined goals. Whereas the zero-shot generalization pushed agents to grow plants with food and water with equal probability, the reward function helped agents to correct that behavior towards more water.

- **Guiding Hindsight Experience Replay (HER).** In multi-goal RL with discrete sets of goals, HER is traditionally used to modify transitions sampled from the replay buffer. It replaces originally targeted goals by others randomly selected from the set of goals Andrychowicz et al. (2017a); **?**. This enables to transfer knowledge between goals, reinterpreting trajectories in the light of new goals. In that case, a reward function is required to compute the reward associated to that new transition (new goal). To improve on random goal replay, we favor goal substitution towards goals that actually match the state and have higher chance of leading to rewards. In IMAGINE, we scan a set of 40 goal candidates for each transition, and select substitute goals that match the scene when possible, with probability $p = 0.5$.

- **Exploring like Go-Explore.** In Go-Explore **?**, agents first reach a goal state, then start exploring from there. We could reproduce that behavior in our IMAGINE agents with our internal reward function. The reward function would scan each state during the trajectory. When the targeted goal is found to be reached, the agent could switch to another goal, add noise on its goal embedding, or increase the exploration noise on actions. This might enable agents to explore sequences of goal-directed behaviors. We leave the study of this mechanism for future work.

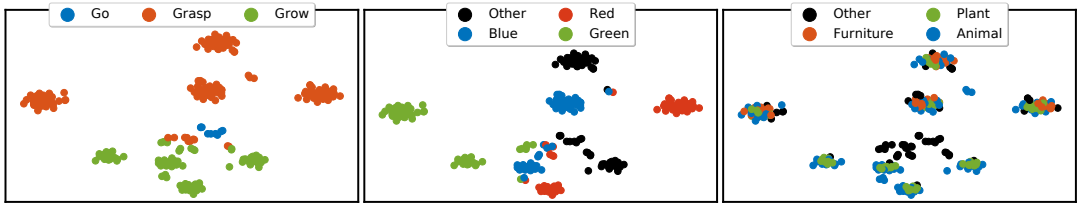
- **Filtering of Imagined Goals.** When generating imagined goals, agents also generate meaningless goals. Ideally, we would like agents to filter these from meaningful goals. Meaningful goals, are goals the agent can interpret with its reward function, goals from which it can learn directed behavior. They are interpreted from known related goals via the generalization of the reward function. If we consider an ensemble of reward functions, chances are that all reward functions in the ensemble will agree on the interpretation of meaningful imagined goals. On the other hand, they might disagree on meaningless goals, as their meanings might not be as easily derived from known related goals. Using an ensemble of reward function may thus help agents filter meaningful goals from meaningless ones. This could be done by labeling a dataset of trajectories with positive or negative rewards and comparing results between reward functions, effectively computing agreement measures for each imagined goals. Having an efficient filtering mechanism would drastically improve the efficiency of goal imagination, as Main Section 8.4.2 showed

that the ratio of meaningful goals determines generalizations performance. This is also left for future work.

D.7 Additional Visualizations

Visualizing Goal Embedding

To analyze the goal embeddings learned by the language encoder L_e, we perform a t-SNE using 2 components, perplexity 20, a learning rate of 10 for 5000 iterations. Figure D.11 presents the resulting projection for a particular run. The embedding seems to be organized mainly in terms of motor predicates (D.11a), then in terms of colors (D.11b). Object types or categories do not seem to be strongly represented (D.11c).



(a) (b) (c)

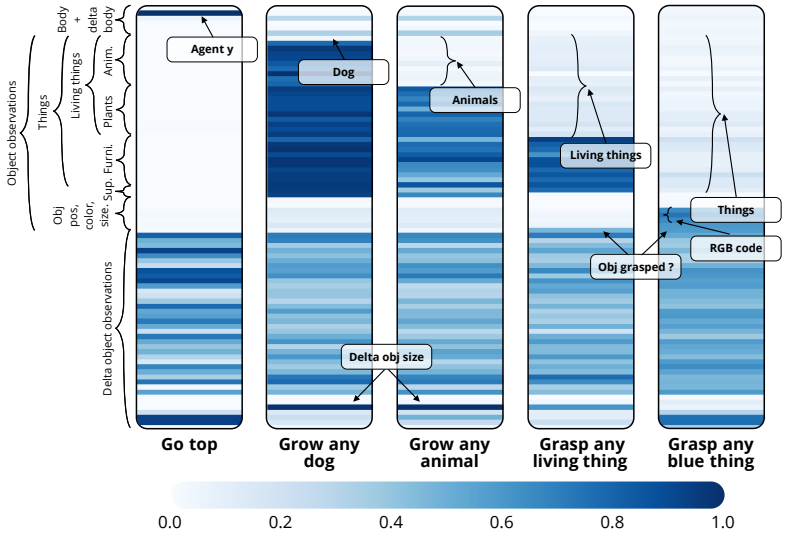
Figure D.11: **t-SNE of Goal Embedding.** The same t-SNE is presented, with different color codes (a) predicates, (b) colors, (c) object categories.

Visualizing Attention Vectors

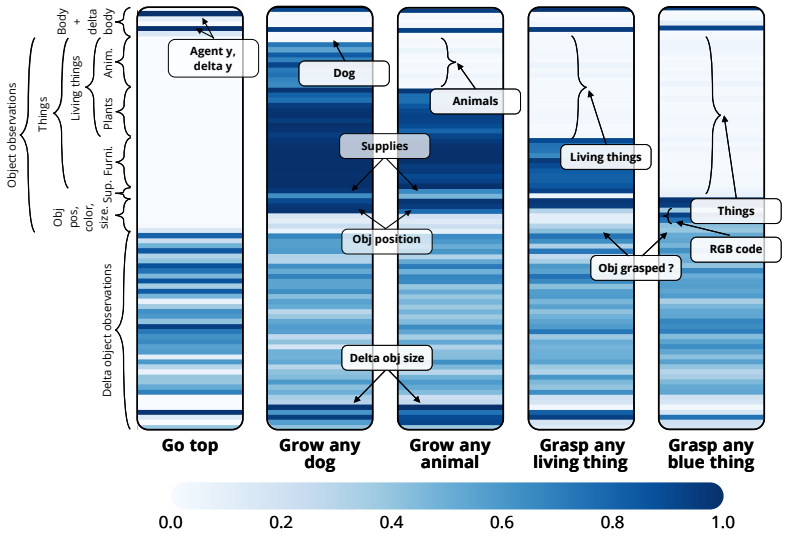
In the *modular-attention* architectures for the reward function and policy, we train attention vectors to be combined with object-specific features using a gated attention mechanism. In each architecture, the attention vector is shared across objects (permutation invariance). Figure D.12 presents examples of attention vectors for the reward function (D.12a) and for the policy (D.12b) at the end of training. These attention vectors highlight relevant parts of the object-specific sub-state depending on the NL goal:

- When the sentence refers to a particular object type (e.g. *dog*) or category (e.g. *living thing*), the attention vector suppresses the corresponding object type(s) and highlights the complement set of object types. If the object does not match the object type or category described in the sentence, the output of the Hadamard product between object types and attention will be close to 1. Conversely, if the object is of the required type, the attention suppression ensures that the output stays close to zero. Although it might not be intuitive for humans, it efficiently detects whether the considered object is the one the sentence refers to.
- When the sentence refers to a navigation goal (e.g. *go top*, the attention highlights the agent's position (here y).
- When the sentence is a *grow* goal, the reward function focuses on the difference in object's size, while the policy further highlights the object's position.

The attention vectors uses information about the goal to highlight or suppress parts of the input using the different strategies described above depending on the type of input (object categories, agent's position, difference in size etc). This type of gated-attention improves the interpretability of the reward function and policy.



(a)



(b)

Figure D.12: **Attention vectors** (a) $\boldsymbol{\alpha}^g$ for the reward function (1 seed). (b) $\boldsymbol{\beta}^g$ for the policy (1 seed).

D.8 Comparing IMAGINE to goal-as-state approaches.

In the goal-conditioned RL literature, some works have proposed goal generation mechanisms to facilitate the acquisition of skills over large sets of goals Nair et al. (2018b); Pong et al. (2020); Colas et al. (2019a); **?**. Some of them had a special interest in exploration, and proposed to bias goal sampling towards goals from low density areas Pong et al. (2020). One might then think that IMAGINE should be compared to these approaches. However, there are a few catches:

1. Nair et al. (2018b); **?**); Pong et al. (2020) use generative models of states to sample state-based goals. However, our environment is procedurally generated. This means that sampling a given state from the generative model has a very low probability to *match* the scene. If the present objects are three red cats, the agent has no chance to reach a goal specifying dogs and lions' positions, colors and sizes. Indeed, most of the state space is made of object features that cannot be acted upon (colors, types, sizes of most objects). One could imagine using SP to organize the scene, but we would need to ask SP to find the three objects specified by the generated goal, in the exact colors (RGB codes) and size. Doing so, there would be no distracting object for agent to discover and learn about. A second option is to condition the goal generation on the scene as it is done in **?**. The question of whether it might work in procedurally-generated environments remains open.

2. Assuming a perfect goal generator that only samples valid goals that do not ask a change of object color or type, the agent would then need to bring each object to its target position and to grow objects to their very specific goal size. These goals are not the same as those targeted by IMAGINE, they are too specific. These approaches –like most goal-conditioned RL approaches– represent goals as particular states (e.g. block positions in manipulation tasks, visual states in navigation tasks) Schaul et al. (2015a); Andrychowicz et al. (2017a); Nair et al. (2018b); Pong et al. (2020); Colas et al. (2019a). In contrast, language-conditioned agents represent abstract goals, usually defined by specific constraints on states (e.g. *grow any plant* requires the size of at least one plant to increase) Chan et al. (2019); Jiang et al. (2019a); Cideron et al. (2020c). For this reason, *goal-as-state* and *abstract goal* approaches do not tackle the same problem. The first targets specific coordinates, and cannot be instructed to reach abstract goals, while the second are not trained to reach specific states.

For these reasons, we argue that the goal-conditioned approaches that use state-based goals cannot be easily or fairly compared to our approach IMAGINE.

D.9 Implementation details

Reward function inputs and hyperparameters.

Supplementary Section D.5 details the architecture of the reward function. The following provides extra details about the inputs. The object-dependent sub-state $\mathbf{s}_{obj(i)}$ contains information about both the agent's body and the corresponding object i: $\mathbf{s}_{obj(i)} = [\mathbf{o}_{body}, \Delta\mathbf{o}_{body}, \mathbf{o}_{obj(i)}, \Delta\mathbf{o}_{obj(i)}]$ where $\mathbf{o}_{body}$ and $\mathbf{o}_{obj(i)}$ are body- and obj_i-dependent observations, and $\Delta\mathbf{o}^t_{body} = \mathbf{o}^t_{body} - \mathbf{o}^0_{body}$ and $\Delta\mathbf{o}^t_{obj(i)} = \mathbf{o}^t_{obj(i)} - \mathbf{o}^0_{obj(i)}$ measure the difference between the initial and current observations. The second input is the attention vector $\boldsymbol{\alpha}^g$ that is integrated with $\mathbf{s}_{obj(i)}$ through an Hadamard product to form the model input: $\mathbf{x}^g_i = \mathbf{s}_{obj(i)} \odot \boldsymbol{\alpha}^g$. This attention vector is a simple mapping from $\mathbf{g}$ to a vector of the size of $\mathbf{s}_{obj(i)}$ contained in $[0,1]^{size(\mathbf{s}_{obj(i)})}$. This cast is implemented by a one-layer neural network with sigmoid activations $\texttt{NN}^{\text{cast}}$ such that $\boldsymbol{\alpha}^g = \texttt{NN}^{\text{cast}}(\mathbf{g})$.

For the three architectures the number of hidden units of the `LSTM` and the sizes of the hidden layers of fully connected networks are fixed to 100. `NN` parameters are initialized using He initialization He et al. (2015a) and we use one-hot word encodings. The `LSTM` is implemented using `rnn.BasicLSTMCell` from tensorflow 1.15 based on Zaremba et al. (2014). The states are initially set to zero. The `LSTM`'s weights are initialized uniformly from $[-0.1, 0.1]$ and the biases initially set to zero. The `LSTM` use a *tanh* activation function whereas the `NN` are using ReLU activation functions in their hidden layers and sigmoids at there output.

Reward function training schedule.

The architecture are trained via backpropagation using the Adam Optimizer Kingma & Ba (2015). The data is fed to the model in batches of 512 examples. Each batch is constructed so that it contains at least one instance of each goal description g_{NL} (goals discovered so far). We also use a modular buffer to impose a ratio of positive rewards of 0.2 for each description in each batch. When trained in parallel of the policy, the reward function is updated once every 1200 episodes. Each update corresponds to up to 100 training epochs (100 batches). We implement a stopping criterion based on the F_1-score computed from a held-out test set uniformly sampled from the last episodes (20% of the last 1200 episodes (2 epochs)). The update is stopped when the F_1-score on the held-out set does not improve for 10 consecutive training epochs.

RL implementation and hyperparameters.

In the policy and critic architectures, we use hidden layers of size 256 and ReLU activations. Attention vectors are cast from goal embeddings using single-layer neural networks with sigmoid activations. We use the He initialization scheme for He et al. (2015a) and train them via backpropagation using the Adam optimizer ($\beta_1 = 0.9, \beta_2 = 0.999$) Kingma & Ba (2015).

Our learning algorithm is built on top of the OpenAI Baselines implementation of HER-DDPG.[1] We leverage a parallel implementation with 6 actors. Actors share the same policy and critic parameters but maintain their own memory and conduct their own updates independently. Updates are then summed to compute the next set of parameters broadcast to all actors. Each actor is updated for 50 epochs with batches of size 256 every 2 episodes of environment interactions. Using hindsight replay, we enforce a ratio $p = 0.5$ of transitions associated with positive rewards in each batch. We use the same hyperparameters as Plappert et al. (2018).

Computing resources.

The RL experiments contain 8 conditions of 10 seeds each, and 4 conditions with 5 seeds (SP study). Each run leverages 6 cpus (6 actors) for about 36h for a total of 2.5 cpu years. Experiments presented in this paper requires machines with at least 6 cpu cores.

[1] The OpenAI Baselines implementation of HER-DDPG can be found at https://github.com/openai/baselines, our implementation can be found at https://sites.google.com/view/imagine-drl.[2]

Bibliography

Abramson, J., Ahuja, A., Brussee, A., Carnevale, F., Cassin, M., Clark, S., Dudzik, A., Georgiev, P., Guy, A., Harley, T., Hill, F., Hung, A., Kenton, Z., Landon, J., Lillicrap, T. P., Mathewson, K., Muldal, A., Santoro, A., Savinov, N., Varma, V., Wayne, G., Wong, N., Yan, C., and Zhu, R.
Imitating Interactive Intelligence.
CoRR, abs/2012.05672, 2020.
_eprint: 2012.05672.

Achiam, J. and Sastry, S.
Surprise-based intrinsic motivation for deep reinforcement learning.
ArXiv - abs/1703.01732, 2017.

Achiam, J., Edwards, H., Amodei, D., and Abbeel, P.
Variational option discovery algorithms.
ArXiv - abs/1807.10299, 2018.

Ahilan, S. and Dayan, P.
Feudal multi-agent hierarchies for cooperative reinforcement learning.
arXiv preprint arXiv:1901.08492, 2019.

Ahn, M., Brohan, A., Brown, N., Chebotar, Y., Cortes, O., David, B., Finn, C., Gopalakrishnan, K., Hausman, K., Herzog, A., Ho, D., Hsu, J., Ibarz, J., Ichter, B., Irpan, A., Jang, E., Ruano, R. J., Jeffrey, K., Jesmonth, S., Joshi, N. J., Julian, R., Kalashnikov, D., Kuang, Y., Lee, K.-H., Levine, S., Lu, Y., Luu, L., Parada, C., Pastor, P., Quiambao, J., Rao, K., Rettinghouse, J., Reyes, D., Sermanet, P., Sievers, N., Tan, C., Toshev, A., Vanhoucke, V., Xia, F., Xiao, T., Xu, P., Xu, S., and Yan, M.
Do As I Can, Not As I Say: Grounding Language in Robotic Affordances.
ArXiv – abs/2204.01691, 2022.

Akakzia, A., Colas, C., Oudeyer, P.-Y., Chetouani, M., and Sigaud, O.
DECSTR: Learning goal-directed abstract behaviors using pre-verbal spatial predicates in intrinsically motivated agents.
In *Proc. of ICLR*, 2021a.

Akakzia, A., Colas, C., Oudeyer, P.-Y., Chetouani, M., and Sigaud, O.

Grounding Language to Autonomously-Acquired Skills via Goal Generation.
In *ICLR 2021 - Ninth International Conference on Learning Representation*, Vienna / Virtual, Austria, May 2021c.
URL https://hal.inria.fr/hal-03121146.

Akakzia, A., Colas, C., Oudeyer, P.-Y., Chetouani, M., and Sigaud, O.
Grounding Language to Autonomously-Acquired Skills Via Goal Generation.
Proc. of ICLR, 2021b.

Alayrac, J.-B., Donahue, J., Luc, P., Miech, A., Barr, I., Hasson, Y., Lenc, K., Mensch, A., Millican, K., Reynolds, M., et al.
Flamingo: a Visual Language Model for Few-Shot Learning.
ArXiv – abs/2204.14198, 2022.

Allen, C. and Bekoff, M.
Species of Mind: The Philosophy and Biology of Cognitive Ethology.
MIT Press, 1999.

Andrychowicz, M., Crow, D., Ray, A., Schneider, J., Fong, R., Welinder, P., McGrew, B., Tobin, J., Abbeel, P., and Zaremba, W.
Hindsight experience replay.
In *Proc. of NeurIPS*, pp. 5048–5058, 2017a.

Andrychowicz, M., Crow, D., Ray, A., Schneider, J., Fong, R., Welinder, P., McGrew, B., Tobin, J., Abbeel, P., and Zaremba, W.
Hindsight Experience Replay.
Proc. of NeurIPS, 2017b.

Arora, A., Kaffee, L.-A., and Augenstein, I.
Probing Pre-Trained Language Models for Cross-Cultural Differences in Values.
ArXiv – abs/2203.13722, 2022.

Asada, M., Hosoda, K., Kuniyoshi, Y., Ishiguro, H., Inui, T., Yoshikawa, Y., Ogino, M., and Yoshida, C.
Cognitive developmental robotics: A survey.
IEEE transactions on autonomous mental development, 1(1):12–34, 2009.

Atance, C. M.
Future thinking in young children.
Current Directions in Psychological Science, 17(4):295–298, 2008.

Bacon, P., Harb, J., and Precup, D.
The option-critic architecture.
In *Proc. of AAAI*, pp. 1726–1734, 2017.

Badia, A. P., Piot, B., Kapturowski, S., Sprechmann, P., Vitvitskyi, A., Guo, Z. D., and Blundell, C.
Agent57: Outperforming the atari human benchmark.
In *Proc. of ICML*, volume 119, pp. 507–517, 2020a.

Badia, A. P., Sprechmann, P., Vitvitskyi, A., Guo, D., Piot, B., Kapturowski, S., Tieleman, O., Arjovsky, M., Pritzel, A., Bolt, A., and Blundell, C.
Never give up: Learning directed exploration strategies.
In *Proc. of ICLR*, 2020b.

Bahdanau, D., Hill, F., Leike, J., Hughes, E., Hosseini, S. A., Kohli, P., and Grefenstette, E.
Learning to understand goal specifications by modelling reward.
In *Proc. of ICLR*, 2019a.

Bahdanau, D., Hill, F., Leike, J., Hughes, E., Kohli, P., and Grefenstette, E.
Learning to understand goal specifications by modelling reward.
In *Proc. of ICLR*, 2019b.

Bahdanau, D., Murty, S., Noukhovitch, M., Nguyen, T. H., de Vries, H., and Courville, A. C.
Systematic generalization: What is required and can it be learned?
In *Proc. of ICLR*, 2019c.

Baker, B., Kanitscheider, I., Markov, T., Wu, Y., Powell, G., McGrew, B., and Mordatch, I.
Emergent tool use from multi-agent autocurricula.
In *International Conference on Learning Representations*, 2020.
URL https://openreview.net/forum?id=SkxpxJBKwS.

Baldassarre, G. and Mirolli, M.
Intrinsically motivated learning in natural and artificial systems.
Springer, 2013.

Bandura, A. and Walters, R. H.
Social learning theory, volume 1.
Englewood cliffs Prentice Hall, 1977.

Baranes, A. and Oudeyer, P.-Y.
Proximo-distal competence based curiosity-driven exploration.
In *Learning, in International Conference on Epigenetic Robotics, Italie. Citeseer*. Citeseer, 2009a.

Baranes, A. and Oudeyer, P.-Y.
R-iac: Robust intrinsically motivated exploration and active learning.
IEEE Transactions on Autonomous Mental Development, 1(3):155–169, 2009b.

Baranes, A. and Oudeyer, P.-Y.
Intrinsically motivated goal exploration for active motor learning in robots: A case study.
In *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 1766–1773. IEEE, 2010.

Baranes, A. and Oudeyer, P.-Y.
Active learning of inverse models with intrinsically motivated goal exploration in robots.
Robotics and Autonomous Systems, 61(1):49–73, 2013.

Barde, P., Karch, T., Nowrouzezahrai, D., Moulin-Frier, C., Pal, C., and Oudeyer, P.-Y.
Learning to guide and to be guided in the architect-builder problem.
In *Proc. of ICLR*, 2022.
URL https://openreview.net/forum?id=swiyAeGzFhQ.

Barreto, A., Dabney, W., Munos, R., Hunt, J. J., Schaul, T., Silver, D., and van Hasselt, H.
Successor features for transfer in reinforcement learning.
In *Proc. of NeurIPS*, pp. 4055–4065, 2017.

Barreto, A., Borsa, D., Hou, S., Comanici, G., Aygün, E., Hamel, P., Toyama, D., hunt, J., Mourad, S., Silver, D., and Precup, D.
The option keyboard: Combining skills in reinforcement learning.
In *Proc. of NeurIPS*, volume 32, 2019.

Barreto, A., Hou, S., Borsa, D., Silver, D., and Precup, D.
Fast reinforcement learning with generalized policy updates.
Proceedings of the National Academy of Sciences, 117(48):30079–30087, 2020.

Barto, A. G. and Simsek, O.
Intrinsic Motivation for Reinforcement Learning Systems.
Proceedings of the Thirteenth Yale Workshop on Adaptive and Learning Systems, 2005.

Battaglia, P. W., Hamrick, J. B., Bapst, V., Sanchez-Gonzalez, A., Zambaldi, V., Malinowski, M., Tacchetti, A., Raposo, D., Santoro, A., Faulkner, R., Gulcehre, C., Song, F., Ballard, A., Gilmer, J., Dahl, G., Vaswani, A., Allen, K., Nash, C., Langston, V., Dyer, C., Heess, N., Wierstra, D., Kohli, P., Botvinick, M., Vinyals, O., Li, Y., and Pascanu, R.
Relational inductive biases, deep learning, and graph networks, 2018.

Bellemare, M. G., Srinivasan, S., Ostrovski, G., Schaul, T., Saxton, D., and Munos, R.
Unifying count-based exploration and intrinsic motivation.
In *Proc. of NeurIPS*, pp. 1471–1479, 2016.

Bellemare, M. G., Candido, S., Castro, P. S., Gong, J., Machado, M. C., Moitra, S., Ponda, S. S., and Wang, Z.
Autonomous navigation of stratospheric balloons using reinforcement learning.
Nature, 588(7836):77–82, 2020.

Bender, E. M., Gebru, T., McMillan-Major, A., and Shmitchell, S.
On the dangers of stochastic parrots: Can language models be too big?
In *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*, pp. 610–623, 2021.

Berk, L. E.
Why Children Talk to Themselves.
Scientific American, 1994.

Berlyne, D. E.
Curiosity and exploration.
Science, 153(3731):25–33, 1966a.

Berlyne, D. E.
Curiosity and Exploration.
Science, 1966b.

Berseth, G., Geng, D., Devin, C., Finn, C., Jayaraman, D., and Levine, S.
Smirl: Surprise minimizing rl in dynamic environments.
arXiv preprint arXiv:1912.05510, 2019.

Bisk, Y., Holtzman, A., Thomason, J., Andreas, J., Bengio, Y., Chai, J., Lapata, M., Lazaridou, A., May, J., Nisnevich, A., Pinto, N., and Turian, J.
Experience grounds language.
In *Proc. of EMNLP*. Association for Computational Linguistics, 2020.

Blaes, S., Pogancic, M. V., Zhu, J., and Martius, G.
Control what you can: Intrinsically motivated task-planning agent.
In *Proc. of NeurIPS*, pp. 12520–12531, 2019.

Bonawitz, E., Shafto, P., Gweon, H., Goodman, N., Spelke, E., and Schulz, L.
The double-edged sword of pedagogy: Instruction limits spontaneous exploration and discovery.
Cognition, 120:322–30, 09 2011.

Branavan, S., Zettlemoyer, L., and Barzilay, R.
Reading between the lines: Learning to map high-level instructions to commands.
In *Proc. of ACL*, pp. 1268–1277. Association for Computational Linguistics, 2010.

Brewer, K., Pollock, N., and Wright, F. V.
Addressing the Challenges of Collaborative Goal Setting with Children and Their Families.
Physical & Occupational Therapy in Pediatrics, 2014.

Brighton, H. and Kirby, S.
Understanding linguistic evolution by visualizing the emergence of topographic mappings.
Artificial Life, 12:229–242, 2006.

Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D. M., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., McCandlish, S., Radford, A., Sutskever, I., and Amodei, D.
Language Models are Few-Shot Learners.
Proc. of NeurIPS, abs/2005.14165, 2020.

Bruner, J.
Child's talk: Learning to use language.
Child Language Teaching and Therapy, 1(1):111–114, 1985.

Bruner, J.
Acts of meaning.
Harvard university press, 1990.

Bruner, J.
The narrative construction of reality.
Critical inquiry, 18(1):1–21, 1991.

Burda, Y., Edwards, H., Storkey, A. J., and Klimov, O.
Exploration by random network distillation.
In *Proc. of ICLR*, 2019.

Burgess, C. P., Matthey, L., Watters, N., Kabra, R., Higgins, I., Botvinick, M., and Lerchner, A.
Monet: Unsupervised scene decomposition and representation.
ArXiv - abs/1901.11390, 2019.

Campero, A., Raileanu, R., Küttler, H., Tenenbaum, J. B., Rocktäschel, T., and Grefenstette, E.
Learning with AMIGo: Adversarially Motivated Intrinsic Goals.
Proc. of ICLR, 2021.

Campos, V., Trott, A., Xiong, C., Socher, R., Giró-i-Nieto, X., and Torres, J.
Explore, discover and learn: Unsupervised discovery of state-covering skills.
In *Proc. of ICML*, volume 119, pp. 1317–1327, 2020.

Cangelosi, A. and Parisi, D.
Simulating the evolution of language.
In *Springer London*, 2002.

Cangelosi, A. and Schlesinger, M.
Developmental robotics: From babies to robots.
MIT press, 2015.

Cangelosi, A., Metta, G., Sagerer, G., Nolfi, S., Nehaniv, C., Fischer, K., Tani, J., Belpaeme, T., Sandini, G., Nori, F., et al.
Integration of action and language knowledge: A roadmap for developmental robotics.
IEEE Transactions on Autonomous Mental Development, 2(3), 2010a.

Cangelosi, A., Metta, G., Sagerer, G., Nolfi, S., Nehaniv, C., Fischer, K., Tani, J., Belpaeme, T., Sandini, G., Nori, F., Fadiga, L., Wrede, B., Rohlfing, K., Tuci, E., Dautenhahn, K., Saunders, J., and Zeschel, A.
Integration of action and language knowledge: A roadmap for developmental robotics.
IEEE Transactions on Autonomous Mental Development, 2(3):167–195, 2010b.
doi: 10.1109/TAMD.2010.2053034.

Cao, K., Lazaridou, A., Lanctot, M., Leibo, J. Z., Tuyls, K., and Clark, S.
Emergent communication through negotiation.
In *International Conference on Learning Representations*, 2018.

Carion, N., Massa, F., Synnaeve, G., Usunier, N., Kirillov, A., and Zagoruyko, S.
End-to-end object detection with transformers, 2020.

Carruthers, P.
Thinking in language?: Evolution and a modularist possibility.
In *Language and Thought*. Cambridge University Press, 1998.

Carruthers, P.
Modularity, Language, and the Flexibility of Thought.
Behavioral and Brain Sciences, (6), 2002.
ISSN 0140-525X, 1469-1825.

Carruthers, P. and Boucher, J.
Language and Thought.
Cambridge University Press, 1998.

Cederborg, T. and Oudeyer, P.-Y.
A social learning formalism for learners trying to figure out what a teacher wants them to do.
Paladyn: Journal of Behavioral Robotics, 5:64–99, 2014.

Celik, O., Zhou, D., Li, G., Becker, P., and Neumann, G.
Specializing Versatile Skill Libraries using Local Mixture of Experts.
In *5th Annual Conference on Robot Learning*, 2021.

Chaabouni, R., Kharitonov, E., Bouchacourt, D., Dupoux, E., and Baroni, M.
Compositionality and generalization in emergent languages.
In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 4427–4442, Online, July 2020. Association for Computational Linguistics.
doi: 10.18653/v1/2020.acl-main.407.
URL https://aclanthology.org/2020.acl-main.407.

Chaabouni, R., Kharitonov, E., Dupoux, E., and Baroni, M.
Communicating artificial neural networks develop efficient color-naming systems.
Proceedings of the National Academy of Sciences of the United States of America, 118, 2021.

Chan, H., Wu, Y., Kiros, J., Fidler, S., and Ba, J.
Actrce: Augmenting experience via teacher's advice for multi-goal reinforcement learning.
ArXiv - abs/1902.04546, 2019.

Chaplot, D. S., Sathyendra, K. M., Pasumarthi, R. K., Rajagopal, D., and Salakhutdinov, R.
Gated-attention architectures for task-oriented language grounding.
In *Proc. of AAAI*, pp. 2819–2826, 2018.

Charlesworth, H. and Montana, G.
Plangan: Model-based planning with sparse rewards and multiple goals.
In *Proc. of NeurIPS*, 2020.

Chen, D. L. and Mooney, R. J.
Learning to interpret natural language navigation instructions from observations.
In *Proc. of AAAI*, 2011.

Chen, T., Kornblith, S., Norouzi, M., and Hinton, G.
A simple framework for contrastive learning of visual representations.

In III, H. D. and Singh, A. (eds.), *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 1597–1607. PMLR, 13–18 Jul 2020.
URL https://proceedings.mlr.press/v119/chen20j.html.

Chen, V., Gupta, A., and Marino, K.
Ask Your Humans: Using Human Instructions to Improve Generalization in Reinforcement Learning.
Proc. of ICLR, 2021.

Cheney, D. L. and Seyfarth, R. M.
Constraints and preadaptations in the earliest stages of language evolution.
2005.

Chevalier-Boisvert, M., Bahdanau, D., Lahlou, S., Willems, L., Saharia, C., Nguyen, T. H., and Bengio, Y.
BabyAI: First Steps Towards Grounded Language Learning With a Human In the Loop.
In *Proc. of ICLR*, 2019a.

Chevalier-Boisvert, M., Bahdanau, D., Lahlou, S., Willems, L., Saharia, C., Nguyen, T. H., and Bengio, Y.
Baby-Ai: First Steps Towards Grounded Language Learning with a Human in the Loop.
Proc. of ICLR, 2019b.

Chevalier-Boisvert, M., Bahdanau, D., Lahlou, S., Willems, L., Saharia, C., Nguyen, T. H., and Bengio, Y.
Babyai: A platform to study the sample efficiency of grounded language learning, 2019c.

Chiang, K.-J., Emmanouilidou, D., Gamper, H., Johnston, D., Jalobeanu, M., Cutrell, E., Wilson, A., An, W. W., and Tashev, I.
A closed-loop adaptive brain-computer interface framework: Improving the classifier with the use of error-related potentials.
In *2021 10th International IEEE/EMBS Conference on Neural Engineering (NER)*, pp. 487–490, 2021.
doi: 10.1109/NER49283.2021.9441133.

Chitnis, R., Silver, T., Tenenbaum, J., Kaelbling, L. P., and Lozano-Pérez, T.
Glib: Efficient exploration for relational model-based reinforcement learning via goal-literal babbling.
In *AAAI*, 2021.

Choi, E., Lazaridou, A., and de Freitas, N.
Multi-agent compositional communication learning from raw visual input.
In *International Conference on Learning Representations*, 2018.
URL https://openreview.net/forum?id=rknt2Be0-.

Choi, J., Sharma, A., Lee, H., Levine, S., and Gu, S. S.
Variational Empowerment as Representation Learning for Goal-Based Reinforcement Learning.
ArXiv - abs/2106.01404, 2021.

Chomsky, N.
Syntactic structures.
Mouton, 1957a.
ISBN 9789027933850.

Chomsky, N.
Syntactic Structures.
Mouton, 1957b.
ISBN 978-90-279-3385-0.

Chopra, S., Tessler, M. H., and Goodman, N. D.
The first crank of the cultural ratchet: Learning and transmitting concepts through language.
In *CogSci*, pp. 226–232, 2019.

Christiansen, M. H. and Kirby, S.
Language evolution: consensus and controversies.
Trends in Cognitive Sciences, 7(7):300–307, 2003.
ISSN 1364-6613.
doi: https://doi.org/10.1016/S1364-6613(03)00136-0.
URL https://www.sciencedirect.com/science/article/pii/S1364661303001360.

Chu, J. and Schulz, L.
Exploratory play, rational action, and efficient search.
In Denison, S., Mack, M., 0023, Y. X., and Armstrong, B. C. (eds.), *Proceedings of the 42th Annual Meeting of the Cognitive Science Society - Developing a Mind: Learning in Humans, Animals, and Machines, CogSci 2020, virtual, July 29 - August 1, 2020*. cognitivesciencesociety.org, 2020.
URL https://cogsci.mindmodeling.org/2020/papers/0169/index.html.

Chua, K., Calandra, R., McAllister, R., and Levine, S.
Deep reinforcement learning in a handful of trials using probabilistic dynamics models.
In *Proc. of NeurIPS*, pp. 4759–4770, 2018.

Cideron, G., Seurin, M., Strub, F., and Pietquin, O.
Higher: Improving instruction following with hindsight generation for experience replay.
In *2020 IEEE Symposium Series on Computational Intelligence (SSCI)*, pp. 225–232, 2020a.
doi: 10.1109/SSCI47803.2020.9308603.

Cideron, G., Seurin, M., Strub, F., and Pietquin, O.
HIGhER: Improving Instruction Following with Hindsight Generation for Experience Replay.
In *IEEE Symposium Series on Computational Intelligence (SSCI)*, 2020b.

Cideron, G., Seurin, M., Strub, F., and Pietquin, O.
Higher: Improving instruction following with hindsight generation for experience replay.
In *2020 IEEE Symposium Series on Computational Intelligence (SSCI)*, pp. 225–232. IEEE, 2020c.

Clark, A.
Magic Words: How Language Augments Human Computation.
In Carruthers, P. and Boucher, J. (eds.), *Language and Thought*. Cambridge University Press, 1 edition, 1998a.
ISBN 978-0-521-63108-2 978-0-521-63758-9 978-0-511-59790-9.

Clark, A.
Being There: Putting Brain, Body, and World Together Again.
MIT press, 1998b.

Co-Reyes, J. D., Gupta, A., Sanjeev, S., Altieri, N., Andreas, J., DeNero, J., Abbeel, P., and Levine, S.
Guiding policies with language via meta-learning.
In *Proc. of ICLR*, 2019.

Codevilla, F., Müller, M., López, A., Koltun, V., and Dosovitskiy, A.
End-to-end driving via conditional imitation learning.
In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1–9. IEEE, 2018.

Colas, C., Sigaud, O., and Oudeyer, P.
GEP-PG: decoupling exploration and exploitation in deep reinforcement learning algorithms.
In *Proc. of ICML*, volume 80, pp. 1038–1047, 2018.

Colas, C., Oudeyer, P., Sigaud, O., Fournier, P., and Chetouani, M.
CURIOUS: intrinsically motivated modular multi-goal reinforcement learning.
In *Proc. of ICML*, volume 97, pp. 1331–1340, 2019a.

Colas, C., Sigaud, O., and Oudeyer, P.-Y.
A hitchhiker's guide to statistical comparisons of reinforcement learning algorithms.
ArXiv - abs/1904.06979, 2019b.

Colas, C., Akakzia, A., Oudeyer, P.-Y., Chetouani, M., and Sigaud, O.
Language-conditioned goal generation: a new approach to language grounding for rl.
ArXiv - abs/2006.07043, 2020a.

Colas, C., Karch, T., Lair, N., Dussoux, J., Moulin-Frier, C., Dominey, P. F., and Oudeyer, P.
Language as a cognitive tool to imagine goals in curiosity driven exploration.
In *Proc. of NeurIPS*, 2020b.

Colas, C., Karch, T., Lair, N., Dussoux, J., Moulin-Frier, C., Dominey, P. F., and Oudeyer, P.
Language as a Cognitive Tool to Imagine Goals in Curiosity Driven Exploration.
Proc. of NeurIPS, 2020c.

Colas, C., Madhavan, V., Huizinga, J., and Clune, J.
Scaling map-elites to deep neuroevolution.
In *Proc. of GECCO*, pp. 67–75, 2020d.

Colas, C., Hejblum, B., Rouillon, S., Thiébaut, R., Oudeyer, P.-Y., Moulin-Frier, C., and Prague, M.
Epidemioptim: A toolbox for the optimization of control policies in epidemiological models.
Journal of Artificial Intelligence Research, 71, 2021a.

Colas, C., Karch, T., Moulin-Frier, C., and Oudeyer, P.-Y.
Language as a Cognitive Tool: Dall-E, Humans and Vygotskian RL Agents, March 2021b.
URL https://hal.archives-ouvertes.fr/hal-03159786.

Colas, C., Karch, T., Sigaud, O., and Oudeyer, P.-Y.
Autotelic Agents with Intrinsically Motivated Goal-conditioned Reinforcement Learning: a Short Survey.
Journal of Artificial Intelligence Research, 2022.

Crawford, V. P. and Sobel, J.
Strategic information transmission.
Econometrica: Journal of the Econometric Society, pp. 1431–1451, 1982.

Creswell, A., Shanahan, M., and Higgins, I.
Selection-Inference: Exploiting Large Language Models for Interpretable Logical Reasoning.
ArXiv – abs/2205.09712, 2022.

Csikzentmihalyi, M.
Finding flow: The psychology of engagement with everyday life.
New York: Basic, 1997.

Côté, M.-A., K\'{a}d\'{a}r, \., Yuan, X., Kybartas, B., Barnes, T., Fine, E., Moore, J., Hausknecht, M. J., Asri, L. E., Adada, M., Tay, W., and Trischler, A.
TextWorld: A Learning Environment for Text-Based Games.
Computer Games - 7th Workshop at IJCAI, 2018.

Dai, S., Xu, W., Hofmann, A., and Williams, B.
An Empowerment-based Solution to Robotic Manipulation Tasks with Sparse Rewards.
ArXiv - abs/2010.07986, 2020.

Das, A., Datta, S., Gkioxari, G., Lee, S., Parikh, D., and Batra, D.
Embodied question answering.
In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pp. 1–10. IEEE Computer Society, 2018b.

Das, A., Datta, S., Gkioxari, G., Lee, S., Parikh, D., and Batra, D.
Embodied question answering.
Proc. of CVPR, 2018a.

Dautenhahn, K. and Billard, A.
Studying Robot Social Cognition Within a Developmental Psychology Framework.
Proc. of Eurobot (IEEE), 1999.

Dautenhahn, K., Ogden, B., and Quick, T.
From Embodied to Socially Embedded Agents – Implications for Interaction-Aware Robots.
Cognitive Systems Research, (3), 2002.
ISSN 13890417.

Dayan, P. and Hinton, G. E.
Feudal reinforcement learning.
In *Advances in Neural Information Processing Systems 5, [NIPS Conference]*, pp. 271–278, San Francisco, CA, USA, 1992. Morgan Kaufmann Publishers Inc.
ISBN 1558602747.

Dayan, P. and Hinton, G. E.
Feudal reinforcement learning.
In *Proc. of NeurIPS*, pp. 271–278, 1993a.

Dayan, P. and Hinton, G. E.
Feudal Reinforcement Learning.
In *Advances in neural information processing systems*, 1993b.

Dayan, P., Hinton, G. E., Neal, R. M., and Zemel, R. S.
The Helmholtz Machine.
Neural Computation, 7(5):889–904, 1995.
ISSN 0899-7667, 1530-888X.

de Boer, B. G.
Self-organization in vowel systems.
J. Phonetics, 28:441–465, 2000.

deBettencourt, M. T., Cohen, J. D., Lee, R. F., Norman, K. A., and Turk-Browne, N. B.
Closed-loop training of attention with real-time brain imaging.
Nature neuroscience, 18:470 – 475, 2015.

Degrave, J., Felici, F., Buchli, J., Neunert, M., Tracey, B., Carpanese, F., Ewalds, T., Hafner, R., Abdolmaleki, A., de Las Casas, D., et al.
Magnetic control of tokamak plasmas through deep reinforcement learning.
Nature, 602(7897):414–419, 2022.

Dennett, D. C.
Consciousness Explained.
Penguin uk, 1993.

Dessalles, J.-L.
Aux origines du langage.
Hermès-science, 2000.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K.
BERT: Pre-training of deep bidirectional transformers for language understanding.
In *Proc. of NAACL-HLT*, pp. 4171–4186. Association for Computational Linguistics, 2019.

Ding, D., Hill, F., Santoro, A., and Botvinick, M.
Object-based attention for spatio-temporal reasoning: Outperforming neuro-symbolic models with flexible distributed architectures, 2020.

Ding, Y., Florensa, C., Abbeel, P., and Phielipp, M.
Goal-conditioned imitation learning.
In *Proc. of NeurIPS*, pp. 15298–15309, 2019.

Dominey, P. F.
Emergence of grammatical constructions: evidence from simulation and grounded agent experiments.
Connection Science, 17(3-4):289–306, 2005.
ISSN 0954-0091.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., and Houlsby, N.
An image is worth 16x16 words: Transformers for image recognition at scale.
In *International Conference on Learning Representations*, 2021.
URL https://openreview.net/forum?id=YicbFdNTTy.

Dove, G.
Language as a Disruptive Technology: Abstract Concepts, Embodiment and the Flexible Mind.
Philosophical Transactions of the Royal Society B: Biological Sciences, 2018.

Ecoffet, A., Huizinga, J., Lehman, J., Stanley, K. O., and Clune, J.
First return, then explore.
Nature, 590(7847):580–586, 2021.

Elliot, A. J. and Fryer, J. W.
The goal construct in psychology.
Handbook of motivation science, 18:235–250, 2008.

Elliott, E. M., Morey, C. C., AuBuchon, A. M., Cowan, N., Jarrold, C., Adams, E. J., Attwood, M., Bayram, B., Beeler-Duden, S., Blakstvedt, T. Y., Büttner, G., Castelain, T., Cave, S., Crepaldi, D., Fredriksen, E., Glass, B. A., Graves, A. J., Guitard, D., Hoehl, S., Hosch, A., Jeanneret, S., Joseph, T. N., Koch, C., Lelonkiewicz, J. R., Lupyan, G., McDonald, A., Meissner, G., Mendenhall, W., Moreau, D., Ostermann, T., Özdoğru, A. A., Padovani, F., Poloczek, S., Röer, J. P., Schonberg, C. C., Tamnes, C. K., Tomasik, M. J., Valentini, B., Vergauwe, E., Vlach, H. A., and Voracek, M.
Multilab Direct Replication of Flavell, Beach, and Chinsky (1966): Spontaneous Verbal Rehearsal in a Memory Task as a Function of Age.
Advances in Methods and Practices in Psychological Science, 4(2), 2021.

Elman, J. L.
Learning and development in neural networks: the importance of starting small.
Cognition, 48(1):71 – 99, 1993.
ISSN 0010-0277.

Engelcke, M., Kosiorek, A. R., Jones, O. P., and Posner, I.
Genesis: Generative scene inference and sampling with object-centric latent representations, 2020.

Etcheverry, M., Moulin-Frier, C., and Oudeyer, P.
Hierarchically organized latent modules for exploratory search in morphogenetic systems.
In *Proc. of NeurIPS*, 2020.

Eysenbach, B., Gupta, A., Ibarz, J., and Levine, S.
Diversity is all you need: Learning skills without a reward function.
In *Proc. of ICLR*, 2019.

Eysenbach, B., Geng, X., Levine, S., and Salakhutdinov, R. R.
Rewriting history with inverse RL: hindsight inference for policy improvement.
In *Proc. of NeurIPS*, 2020.

Fan, L., Wang, G., Jiang, Y., Mandlekar, A., Yang, Y., Zhu, H., Tang, A., Huang, D.-A., Zhu, Y., and Anandkumar, A.
Minedojo: Building open-ended embodied agents with internet-scale knowledge.
Proc. of AAAI, 2022.

Fang, K., Zhu, Y., Savarese, S., and Fei-Fei, L.
Discovering Generalizable Skills via Automated Generation of Diverse Tasks.
In *Proceedings of Robotics: Science and Systems*, 2021.

Florensa, C., Held, D., Geng, X., and Abbeel, P.
Automatic goal generation for reinforcement learning agents.
In *Proc. of ICML*, volume 80, pp. 1514–1523, 2018.

Florensa, C., Degrave, J., Heess, N., Springenberg, J. T., and Riedmiller, M.
Self-supervised learning of image embedding for continuous control.
ArXiv - abs/1901.00943, 2019.

Fodor, J. A.
The Language of Thought.
Harvard university press, 1975.

Foerster, J. N., Assael, Y., de Freitas, N., and Whiteson, S.
Learning to communicate with deep multi-agent reinforcement learning.
In *Proc. of NeurIPS*, 2016.

Forestier, S. and Oudeyer, P.-Y.
Modular active curiosity-driven discovery of tool use.
In *Intelligent Robots and Systems (IROS), 2016 IEEE/RSJ International Conference on*, pp. 3965–3972. IEEE, 2016.

Forestier, S., Portelas, R., Mollard, Y., and Oudeyer, P.-Y.
Intrinsically motivated goal exploration processes with automatic curriculum learning.
Journal of Machine Learning Research, 23(152):1–41, 2022.
URL http://jmlr.org/papers/v23/21-0808.html.

Fournier, P., Sigaud, O., Chetouani, M., and Oudeyer, P.-Y.
Accuracy-based curriculum learning in deep reinforcement learning.
ArXiv - abs/1806.09614, 2018.

Fournier, P., Colas, C., Chetouani, M., and Sigaud, O.
Clic: Curriculum learning and imitation for object control in nonrewarding environments.
IEEE Transactions on Cognitive and Developmental Systems, 13(2):239–248, 2021.
doi: 10.1109/TCDS.2019.2933371.

Frans, K., Ho, J., Chen, X., Abbeel, P., and Schulman, J.
Meta learning shared hierarchies.
In *Proc. of ICLR*, 2018.

Fu, J., Korattikara, A., Levine, S., and Guadarrama, S.
From language to goals: Inverse reinforcement learning for vision-based instruction following.
In *Proc. of ICLR*, 2019.

Fujimoto, S., Meger, D., and Precup, D.
Off-policy deep reinforcement learning without exploration.
In *Proc. of ICML*, volume 97, pp. 2052–2062, 2019.

Galantucci, B. and Garrod, S.
Experimental semiotics: a review.
Frontiers in human neuroscience, 5:11, 2011.

Gentner, D. and Hoyos, C.
Analogy and Abstraction.
Topics in Cognitive Science, (3), 2017.
ISSN 17568757.

Gentner, D. and Loewenstein, J.
Relational Language and Relational Thought.
Erlbaum, 2002.

Gibson, J. J. J. J.
The senses considered as perceptual systems.
Allen & Unwin, London, 1968.

Glenberg, A. M. and Kaschak, M. P.
Grounding language in action.
Psychonomic Bulletin & Review, 9(3):558–565, 2002a.
ISSN 1069-9384.

Glenberg, A. M. and Kaschak, M. P.
Grounding language in action.
Psychonomic Bulletin & Review, 9(3):558–565, September 2002b.
ISSN 1531-5320.
doi: 10.3758/BF03196313.
URL https://doi.org/10.3758/BF03196313.

Goldberg, A. E.
The Emergence of the Semantics of Argument Structure Constructions.
In *The emergence of language*. Psychology Press, 1999.

Goldberg, A. E.
Constructions: A new theoretical approach to language.
Trends in cognitive sciences, 7(5):219–224, 2003.

Goodfellow, I. J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A. C., and Bengio, Y.
Generative adversarial nets.
In *Proc. of NeurIPS*, pp. 2672–2680, 2014.

Goodrich, M. A. and Schultz, A. C.
Human-robot interaction: a survey.
Now Publishers Inc, 2008.

Gopnik, A., Meltzoff, A. N., and Kuhl, P. K.
The scientist in the crib: Minds, brains, and how children learn.
William Morrow & Co, 1999.

Gottlieb, J. and Oudeyer, P.-Y.
Towards a neuroscience of active sampling and curiosity.
Nature Reviews Neuroscience, 19(12):758–770, 2018a.

Gottlieb, J. and Oudeyer, P.-Y.
Towards a Neuroscience of Active Sampling and Curiosity.
Nature Reviews Neuroscience, 2018b.

Goyal, A., Lamb, A., Hoffmann, J., Sodhani, S., Levine, S., Bengio, Y., and Schölkopf, B.
Recurrent independent mechanisms.
In *Proc. of ICLR*, 2021.

Goyal, P., Niekum, S., and Mooney, R. J.
Using natural language for reward shaping in reinforcement learning.
In *Proc. of IJCAI*, pp. 2385–2391, 2019.

Green, E. J. and Quilty-Dunn, J.
What is an object file?
The British Journal for the Philosophy of Science, 2017.

Greff, K., Kaufman, R. L., Kabra, R., Watters, N., Burgess, C., Zoran, D., Matthey, L., Botvinick, M., and Lerchner, A.
Multi-object representation learning with iterative variational inference.
In *Proc. of ICML*, volume 97, pp. 2424–2433, 2019.

Greff, K., Kaufman, R. L., Kabra, R., Watters, N., Burgess, C., Zoran, D., Matthey, L., Botvinick, M., and Lerchner, A.
Multi-object representation learning with iterative variational inference, 2020.

Gregor, K., Rezende, D. J., and Wierstra, D.
Variational intrinsic control.
ArXiv - abs/1611.07507, 2016.

Grizou, J., Lopes, M., and Oudeyer, P.-Y.
Robot learning simultaneously a task and how to interpret human instructions.
In *2013 IEEE Third Joint International Conference on Development and Learning and Epigenetic Robotics (ICDL)*, pp. 1–8, 2013.
doi: 10.1109/DevLrn.2013.6652523.

Grizou, J., Iturrate, I., Montesano, L., Oudeyer, P.-Y., and Lopes, M.
Calibration-Free BCI Based Control.
In *Twenty-Eighth AAAI Conference on Artificial Intelligence*, pp. 1–8, Quebec, Canada, July 2014.
URL https://hal.archives-ouvertes.fr/hal-00984068.

Gupta, A., Resnick, C., Foerster, J., Dai, A., and Cho, K.
Compositionality and capacity in emergent languages.
In *Proceedings of the 5th Workshop on Representation Learning for NLP*, pp. 34–38, Online, July 2020. Association for Computational Linguistics.
doi: 10.18653/v1/2020.repl4nlp-1.5.
URL https://aclanthology.org/2020.repl4nlp-1.5.

Hadfield-Menell, D., Russell, S. J., Abbeel, P., and Dragan, A.
Cooperative inverse reinforcement learning.
Advances in neural information processing systems, 29:3909–3917, 2016.

Hamrick, J. B., Friesen, A. L., Behbahani, F., Guez, A., Viola, F., Witherspoon, S., Anthony, T., Buesing, L. H., Velickovic, P., and Weber, T.
On the role of planning in model-based deep reinforcement learning.
In *Proc. of ICLR*, 2021.

Harari, Y. N.
Sapiens: A Brief History of Humankind.
Random House, 2014.

Hartikainen, K., Geng, X., Haarnoja, T., and Levine, S.
Dynamical distance learning for semi-supervised and unsupervised skill discovery.
In *Proc. of ICLR*, 2020.

Hausman, K., Springenberg, J. T., Wang, Z., Heess, N., and Riedmiller, M. A.
Learning an embedding space for transferable robot skills.
In *Proc. of ICLR*, 2018.

Havrylov, S. and Titov, I.
Emergence of language with multi-agent games: Learning to communicate with sequences of symbols.
In Guyon, I., Luxburg, U. V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.
URL https://proceedings.neurips.cc/paper/2017/file/70222949cc0db89ab32c9969754d4758-Paper.pdf.

He, K., Zhang, X., Ren, S., and Sun, J.
Delving deep into rectifiers: Surpassing human-level performance on imagenet classification.
In *Proceedings of the IEEE international conference on computer vision*, pp. 1026–1034, 2015a.

He, K., Zhang, X., Ren, S., and Sun, J.
Delving deep into rectifiers: Surpassing human-level performance on imagenet classification.
2015 IEEE International Conference on Computer Vision (ICCV), pp. 1026–1034, 2015b.

He, K., Zhang, X., Ren, S., and Sun, J.
Deep residual learning for image recognition.
2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 770–778, 2016.

Henrich, J. and McElreath, R.
The evolution of cultural evolution.
Evolutionary Anthropology: Issues, News, and Reviews., 2003.

Hermann, K. M., Hill, F., Green, S., Wang, F., Faulkner, R., Soyer, H., Szepesvari, D., Czarnecki, W. M., Jaderberg, M., Teplyashin, D., Wainwright, M., Apps, C., Hassabis, D., and Blunsom, P.
Grounded Language Learning in a Simulated 3D World.
ArXiv - abs/1706.06551, 2017.

Hermer-Vazquez, L.
Language, Space, and the Development of Cognitive Flexibility in Humans: The Case of Two Spatial Memory Tasks.
Cognition, (3), 2001.
ISSN 00100277.

Hershcovich, D., Frank, S., Lent, H., de Lhoneux, M., Abdou, M., Brandl, S., Bugliarello, E., Piqueras, L. C., Chalkidis, I., Cui, R., Fierro, C., Margatina, K., Rust, P., and Søgaard, A.
Challenges and Strategies in Cross-Cultural NLP.
Proc. of ACL, 2022.

Hesse, M.
The Cognitive Claims of Metaphor.
The journal of speculative philosophy, 1988.

Hester, T., Vecerik, M., Pietquin, O., Lanctot, M., Schaul, T., Piot, B., Horgan, D., Quan, J., Sendonaris, A., Osband, I., et al.
Deep q-learning from demonstrations.
In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

Hill, F., Lampinen, A., Schneider, R., Clark, S., Botvinick, M., McClelland, J. L., and Santoro, A.
Emergent systematic generalization in a situated agent.
In *Proc. of ICLR*, 2020a.

Hill, F., Lampinen, A., Schneider, R., Clark, S., Botvinick, M., McClelland, J. L., and Santoro, A.
Environmental drivers of systematicity and generalization in a situated agent, 2020b.

Hill, F., Lampinen, A. K., Schneider, R., Clark, S., Botvinick, M., McClelland, J. L., and Santoro, A.
Emergent Systematic Generalization in a Situated Agent.
Proc. of ICLR, 2020c.

Hill, F., Mokra, S., Wong, N., and Harley, T.
Human Instruction-Following with Deep Reinforcement Learning via Transfer-Learning from Text.
ArXiv – abs/2005.09382, 2020d.

Hill, F., Tieleman, O., von Glehn, T., Wong, N., Merzic, H., and Clark, S.
Grounded language learning fast and slow.
In *Proc. of ICLR*, 2021.

Hinaut, X. and Dominey, P. F.
Real-time parallel processing of grammatical structure in the fronto-striatal system: A recurrent network simulation study using reservoir computing.
PloS one, 8(2), 2013.

Hinaut, X., Petit, M., Pointeau, G., and Dominey, P. F.
Exploring the acquisition and production of grammatical constructions through human-robot interaction with echo state networks.
Frontiers in neurorobotics, 8:16, 2014.

Hintze, A.
Open-Endedness for the Sake of Open-Endedness.
Artificial Life, 25(2):198–206, 2019.
ISSN 1064-5462, 1530-9185.

Ho, J. and Ermon, S.
Generative adversarial imitation learning.
In *Proc. of NeurIPS*, pp. 4565–4573, 2016.

Hochreiter, S. and Schmidhuber, J.
Long short-term memory.
Neural Comput., 9(8):1735–1780, November 1997.
ISSN 0899-7667.
doi: 10.1162/neco.1997.9.8.1735.
URL https://doi.org/10.1162/neco.1997.9.8.1735.

Hockett, C. F. and Hockett, C. D.
The origin of speech.
Scientific American, 203(3):88–97, 1960.

Hoffmann, T.
Construction Grammar and Creativity: Evolution, Psychology, and Cognitive Science.
Cognitive Semiotics, (1), 2020.
ISSN 2235-2066, 1662-1425.

Houthooft, R., Chen, X., Duan, Y., Schulman, J., Turck, F. D., and Abbeel, P.
VIME: variational information maximizing exploration.
In *Proc. of NeurIPS*, pp. 1109–1117, 2016.

Huang, W., Abbeel, P., Pathak, D., and Mordatch, I.
Language models as zero-shot planners: Extracting actionable knowledge for embodied agents.
Proc. of ICML, 2022.

Hui, D. Y.-T., Chevalier-Boisvert, M., Bahdanau, D., and Bengio, Y.
Babyai 1.1, 2020.

Hupkes, D., Dankers, V., Mul, M., and Bruni, E.
Compositionality decomposed: how do neural networks generalise?, 2020.

Jaques, N., Lazaridou, A., Hughes, E., Gulcehre, C., Ortega, P. A., Strouse, D. J., Leibo, J., and de Freitas, N.
Social Influence as Intrinsic Motivation for Multi-Agent Deep Reinforcement Learning.
Proc. of ICML, 2019.

Jiang, Y., Gu, S., Murphy, K., and Finn, C.
Language as an abstraction for hierarchical deep reinforcement learning.
In *Proc. of NeurIPS*, pp. 9414–9426, 2019a.

Jiang, Y., Gu, S., Murphy, K., and Finn, C.
Language as an abstraction for hierarchical deep reinforcement learning.
Proc. of NeurIPS, 2019b.

Johnson, J., Hariharan, B., van der Maaten, L., Fei-Fei, L., Zitnick, C. L., and Girshick, R.
Clevr: A diagnostic dataset for compositional language and elementary visual reasoning, 2016.

Johnson, S. P., Amso, D., and Slemmer, J. A.
Development of object concepts in infancy: Evidence for early learning in an eye-tracking paradigm.
Proceedings of the National Academy of Sciences, 100(18):10568–10573, 2003.

Kaelbling, L. P.
Learning to achieve goals.
In *IJCAI*, pp. 1094–1099. Citeseer, 1993.

Kaplan, F. and Oudeyer, P.-Y.
Maximizing Learning Progress: An Internal Reward System for Development.
In *Embodied artificial intelligence*, pp. 259–270. Springer, 2004.

Kaplan, F. and Oudeyer, P.-Y.
In search of the neural circuits of intrinsic motivation.
Frontiers in neuroscience, 1:17, 2007.

Karch, T., Teodorescu, L., Hofmann, K., Moulin-Frier, C., and Oudeyer, P.-Y.
Grounding Spatio-Temporal Language with Transformers.
Proc. of NeurIPS, 2021.

Katyal, K. D., Johannes, M. S., Kellis, S., Aflalo, T., Klaes, C., McGee, T. G., Para, M. P., Shi, Y., Lee, B., Pejsa, K., Liu, C., Wester, B. A., Tenore, F., Beaty, J. D., Ravitz, A. D., Andersen, R. A., and McLoughlin, M. P.
A collaborative bci approach to autonomous control of a prosthetic limb system.
In *2014 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, pp. 1479–1482, 2014.
doi: 10.1109/SMC.2014.6974124.

Kidd, C. and Hayden, B. Y.
The psychology and neuroscience of curiosity.
Neuron, 88(3):449–460, 2015a.

Kidd, C. and Hayden, B. Y.
The Psychology and Neuroscience of Curiosity.
Neuron, 2015b.

Kim, K., Sano, M., Freitas, J. D., Haber, N., and Yamins, D.
Active world model learning with progress curiosity.
In *Proc. of ICML*, volume 119, pp. 5306–5315, 2020.

Kingma, D. P. and Ba, J.
Adam: A method for stochastic optimization.
In *Proc. of ICLR*, 2015.

Kingma, D. P. and Ba, J.
Adam: A method for stochastic optimization, 2017.

Kirby, S., Griffiths, T., and Smith, K.
Iterated learning and the evolution of language.
Current Opinion in Neurobiology, 28:108–114, 2014.

Kiseleva, J., Li, Z., Aliannejadi, M., Mohanty, S., ter Hoeve, M., Burtsev, M., Skrynnik, A., Zholus, A., Panov, A., Srinet, K., et al.
Neurips 2021 competition iglu: Interactive grounded language understanding in a collaborative environment.
arXiv preprint arXiv:2110.06536, 2021.

Kocsis, L. and Szepesvári, C.
Bandit based monte-carlo planning.
In *European conference on machine learning*, pp. 282–293. Springer, 2006.

Kottur, S., Moura, J. M. F., Lee, S., and Batra, D.
Natural language does not emerge 'naturally' in multi-agent dialog.
In *EMNLP*, 2017.

Kovač, G., Laversanne-Finot, A., and Oudeyer, P.-Y.
Grimgep: Learning progress for robust goal sampling in visual deep reinforcement learning.
ArXiv - abs/2008.04388, 2020.

Krizhevsky, A., Sutskever, I., and Hinton, G. E.
Imagenet classification with deep convolutional neural networks.
Communications of the ACM, 60:84–90, 2012.

Kulkarni, T. D., Narasimhan, K., Saeedi, A., and Tenenbaum, J.
Hierarchical deep reinforcement learning: Integrating temporal abstraction and intrinsic motivation.
In *Proc. of NeurIPS*, pp. 3675–3683, 2016.

Kumar, S., Kumar, A., Levine, S., and Finn, C.
One solution is not all you need: Few-shot extrapolation via structured maxent RL.
In *Proc. of NeurIPS*, 2020.

Lake, B. M. and Baroni, M.
Generalization without systematicity: On the compositional skills of sequence-to-sequence recurrent networks, 2018.

Lakoff, G. and Johnson, M.
Metaphors We Live By.
University of Chicago press, 2008.

Lampinen, A. K., Roy, N. A., Dasgupta, I., Chan, S. C. Y., Tam, A. C., McClelland, J. L., Yan, C., Santoro, A., Rabinowitz, N. C., Wang, J. X., and Hill, F.
Tell Me Why! – Explanations Support Learning of Relational and Causal Structure.
Proc. of ICML, 2022.

Lanier, J. B., McAleer, S., and Baldi, P.
Curiosity-driven multi-criteria hindsight experience replay.
ArXiv - abs/1906.03710, 2019.

Laskin, M., Wang, L., Oh, J., Parisotto, E., Spencer, S., Steigerwald, R., Strouse, D., Hansen, S., Filos, A., Brooks, E., Gazeau, M., Sahni, H., Singh, S., and Mnih, V.
In-context reinforcement learning with algorithm distillation, 2022.
URL https://arxiv.org/abs/2210.14215.

Laversanne-Finot, A., Pere, A., and Oudeyer, P.-Y.
Curiosity driven exploration of learned disentangled goal spaces.
In *Conference on Robot Learning*, pp. 487–504. PMLR, 2018.

Lazaridou, A. and Baroni, M.
Emergent multi-agent communication in the deep learning era.
ArXiv, abs/2006.02419, 2020.

Lazaridou, A., Peysakhovich, A., and Baroni, M.
Multi-agent cooperation and the emergence of (natural) language.
In *International Conference on Learning Representations*, 2017.
URL https://openreview.net/forum?id=Hk8N3Sclg.

Lazaridou, A., Hermann, K. M., Tuyls, K., and Clark, S.
Emergence of linguistic communication from referential games with symbolic and pixel input.
In *International Conference on Learning Representations*, 2018.
URL https://openreview.net/forum?id=HJGv1Z-AW.

LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P.
Gradient-based learning applied to document recognition.
Proc. IEEE, 86:2278–2324, 1998.

Lehman, J. and Stanley, K. O.
Evolving a diversity of virtual creatures through novelty search and local competition.
In *Proceedings of the 13th annual conference on Genetic and evolutionary computation*, pp. 211–218, 2011.

Levy, A., Platt, R., and Saenko, K.
Hierarchical reinforcement learning with hindsight.
ArXiv - abs/1805.08180, 2018.

Li, F. and Bowling, M.
Ease-of-teaching and language structure from emergent communication.
In Wallach, H., Larochelle, H., Beygelzimer, A., d Alché-Buc, F., Fox, E., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.
URL https://proceedings.neurips.cc/paper/2019/file/b0cf188d74589db9b23d5d277238a929-Paper.pdf.

Li, R., Jabri, A., Darrell, T., and Agrawal, P.
Towards practical multi-object manipulation using relational reinforcement learning.
In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 4051–4058. IEEE, 2020.

Liang, P. P., Wu, C., Morency, L.-P., and Salakhutdinov, R.
Towards understanding and mitigating social biases in language models.
In Meila, M. and Zhang, T. (eds.), *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 6565–6576. PMLR, 18–24 Jul 2021.
URL https://proceedings.mlr.press/v139/liang21a.html.

Lillicrap, T. P., Hunt, J. J., Pritzel, A., Heess, N., Erez, T., Tassa, Y., Silver, D., and Wierstra, D.
Continuous control with deep reinforcement learning.
In *Proc. of ICLR*, 2016.

Lindblom, J. and Ziemke, T.
Social Situatedness of Natural and Artificial Intelligence: Vygotsky and Beyond.
Adaptive Behavior, (2), 2003.
ISSN 1059-7123, 1741-2633.

Linke, C., Ady, N. M., White, M., Degris, T., and White, A.
Adapting behavior via intrinsic reward: a survey and empirical study.
Journal of Artificial Intelligence Research, 69:1287–1332, 2020.

Littman, M. L.
Markov games as a framework for multi-agent reinforcement learning.
In Cohen, W. W. and Hirsh, H. (eds.), *Machine Learning Proceedings 1994*, pp. 157–163. Morgan Kaufmann, San Francisco (CA), 1994.
ISBN 978-1-55860-335-6.
doi: https://doi.org/10.1016/B978-1-55860-335-6.50027-1.
URL https://www.sciencedirect.com/science/article/pii/B9781558603356500271.

Locatello, F., Weissenborn, D., Unterthiner, T., Mahendran, A., Heigold, G., Uszkoreit, J., Dosovitskiy, A., and Kipf, T.
Object-centric learning with slot attention, 2020.

Lonini, L., Forestier, S., Teulière, C., Zhao, Y., Shi, B. E., and Triesch, J.
Robust active binocular vision through intrinsically motivated learning.
Frontiers in neurorobotics, 7:20, 2013.

Lopes, M., Lang, T., Toussaint, M., and Oudeyer, P.
Exploration in model-based reinforcement learning by empirically estimating learning progress.
In *Proc. of NeurIPS*, pp. 206–214, 2012.

Lowe, R., WU, Y., Tamar, A., Harb, J., Pieter Abbeel, O., and Mordatch, I.
Multi-agent actor-critic for mixed cooperative-competitive environments.
Proc. of NeurIPS, 30:6379–6390, 2017.

Loynd, R., Fernandez, R., Çelikyilmaz, A., Swaminathan, A., and Hausknecht, M. J.
Working memory graphs.
In *Proc. of ICML*, volume 119, pp. 6404–6414, 2020.

Luketina, J., Nardelli, N., Farquhar, G., Foerster, J. N., Andreas, J., Grefenstette, E., Whiteson, S., and Rocktäschel, T.
A survey of reinforcement learning informed by natural language.
In *Proc. of IJCAI*, pp. 6309–6317, 2019.

Lupyan, G.
Carving Nature at Its Joints and Carving Joints into Nature: How Labels Augment Category Representations.
In *Modeling Language, Cognition and Action*. World Scientific, 2005.
ISBN 978-981-256-324-8 978-981-270-188-6.

Lupyan, G.
What Do Words Do? Toward a Theory of Language-Augmented Thought.
In *Psychology of Learning and Motivation*. Elsevier, 2012.

Lynch, C. and Sermanet, P.
Grounding language in play.
ArXiv - abs/2005.07648, 2020.

Lynch, C. and Sermanet, P.
Language conditioned imitation learning over unstructured data.
Robotics: Science and Systems XVII, 2021.

Lynch, C., Khansari, M., Xiao, T., Kumar, V., Tompson, J., Levine, S., and Sermanet, P.
Learning latent plans from play.
In *Proceedings of the Conference on Robot Learning*, volume 100, pp. 1113–1132, 2020.

Madden, C., Hoen, M., and Dominey, P. F.
A cognitive neuroscience perspective on embodied language for human–robot cooperation.
Brain and Language, 112(3):180–188, mar 2010.
ISSN 0093-934X.
doi: 10.1016/J.BANDL.2009.07.001.

Mangin, O., Filliat, D., Ten Bosch, L., and Oudeyer, P.-Y.
Mca-nmf: Multimodal concept acquisition with non-negative matrix factorization.
PloS one, 10(10):e0140732, 2015.

Mankowitz, D. J., Žídek, A., Barreto, A., Horgan, D., Hessel, M., Quan, J., Oh, J., van Hasselt, H., Silver, D., and Schaul, T.
Unicorn: Continual learning with a universal, off-policy agent.
ArXiv - abs/1802.08294, 2018.

Martius, G., Der, R., and Ay, N.
Information driven self-organization of complex robotic behaviors.
PloS one, 8(5):e63400, 2013.

Marzoev, A., Madden, S., Kaashoek, M. F., Cafarella, M., and Andreas, J.
Unnatural language processing: Bridging the gap between synthetic and natural language data, 2020.

McClelland, J. L., Botvinick, M. M., Noelle, D. C., Plaut, D. C., Rogers, T. T., Seidenberg, M. S., and Smith, L. B.
Letting structure emerge: connectionist and dynamical systems approaches to cognition.
Trends in cognitive sciences, 14(8):348–356, 2010.

McClelland, J. L., Hill, F., Rudolph, M., Baldridge, J., and Schütze, H.
Placing language in an integrated understanding system: Next steps toward human-level performance in neural language models.
Proceedings of the National Academy of Sciences, 117(42):25966–25974, 2020.

McDowell, J.
Mind and World.
Harvard University Press, 1996.

McGovern, A. and Barto, A. G.
Automatic discovery of subgoals in reinforcement learning using diverse density.
In *Proc. of ICML*, pp. 361–368, 2001.

Mihai, D. and Hare, J. S.
Differentiable drawing and sketching.
ArXiv, abs/2103.16194, 2021a.

Mihai, D. and Hare, J. S.
Learning to draw: Emergent communication through sketching.
NeurIPS, 2021b.

Mintz, T. H.
Frequent frames as a cue for grammatical categories in child directed speech.
Cognition, 90(1):91–117, 2003.

Mirchandani, S., Karamcheti, S., and Sadigh, D.
ELLA: Exploration through Learned Language Abstraction.
In Ranzato, M., Beygelzimer, A., Dauphin, Y., Liang, P., and Vaughan, J. W. (eds.), *Advances in Neural Information Processing Systems*, volume 34, pp. 29529–29540. Curran Associates, Inc., 2021.

Mirolli, M. and Parisi, D.
Towards a Vygotskyan Cognitive Robotics: The Role of Language as a Cognitive Tool.
New Ideas in Psychology, 29(3), 2011.
ISSN 0732118X.

Mishra, J. and Gazzaley, A.
Closed-loop cognition: the next frontier arrives.
Trends in Cognitive Sciences, 19:242–243, 2015.

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M., Fidjeland, A. K., Ostrovski, G., et al.
Human-level control through deep reinforcement learning.
nature, 518(7540):529–533, 2015.

Moerland, T. M.
The Intersection of Planning and Learning.
PhD thesis, Delft University of Technology, Netherlands, 2021.

Mohamed, S. and Rezende, D. J.
Variational information maximisation for intrinsically motivated reinforcement learning.
In *Proc. of NeurIPS*, pp. 2125–2133, 2015.

Mordatch, I. and Abbeel, P.
Emergence of grounded compositional language in multi-agent populations.
In *AAAI*, 2018.

Morgan, T. J., Uomini, N. T., Rendell, L. E., Chouinard-Thuly, L., Street, S. E., Lewis, H. M., Cross, C. P., Evans, C., Kearney, R., de la Torre, I., et al.
Experimental evidence for the co-evolution of hominin tool-making teaching and language.
Nature communications, 6(1):1–8, 2015.

Moulin-Frier, C. and Oudeyer, P.-Y.
Multi-agent reinforcement learning as a computational tool for language evolution research: Historical context and future challenges.
ArXiv, abs/2002.08878, 2020.

Moulin-Frier, C., Nguyen, S. M., and Oudeyer, P.-Y.
Self-organization of early vocal development in infants and machines: The role of intrinsic motivation.
Frontiers in Psychology (Cognitive Science), 4(1006), 2014.
ISSN 1664-1078.

Moulin-Frier, C., Diard, J., Schwartz, J.-L., and Bessière, P.
Cosmo ("communicating about objects using sensory–motor operations"): A bayesian modeling framework for studying speech communication and the emergence of phonological systems.
Journal of Phonetics, 53:5–41, 2015.
ISSN 0095-4470.
doi: https://doi.org/10.1016/j.wocn.2015.06.001.
URL https://www.sciencedirect.com/science/article/pii/S0095447015000352.
On the cognitive nature of speech sound systems.

Mouret, J.-B. and Clune, J.
Illuminating search spaces by mapping elites.
arXiv preprint arXiv:1504.04909, 2015.

Mu, J. and Goodman, N.
Emergent communication of generalizations.
In Ranzato, M., Beygelzimer, A., Dauphin, Y., Liang, P., and Vaughan, J. W. (eds.), *Advances in Neural Information Processing Systems*, volume 34, pp. 17994–18007. Curran Associates, Inc., 2021.

URL https://proceedings.neurips.cc/paper/2021/file/9597353e41e6957b5e7aa79214fcb256-Paper.pdf.

Mu, J., Zhong, V., Raileanu, R., Jiang, M., Goodman, N. D., Rocktaschel, T., and Grefenstette, E.
Improving Intrinsic Exploration with Language Abstractions.
ArXiv – abs/2202.08938, 2022.

Muñoz-Moldes, S. and Cleeremans, A.
Delineating implicit and explicit processes in neurofeedback learning.
Neuroscience & Biobehavioral Reviews, 118:681–688, 2020.
ISSN 0149-7634.
doi: https://doi.org/10.1016/j.neubiorev.2020.09.003.
URL https://www.sciencedirect.com/science/article/pii/S0149763420305595.

Nachum, O., Gu, S., Lee, H., and Levine, S.
Data-efficient hierarchical reinforcement learning.
In *Proc. of NeurIPS*, pp. 3307–3317, 2018.

Nair, A., McGrew, B., Andrychowicz, M., Zaremba, W., and Abbeel, P.
Overcoming exploration in reinforcement learning with demonstrations.
In *2018 IEEE international conference on robotics and automation (ICRA)*, pp. 6292–6299. IEEE, 2018a.

Nair, A., Pong, V., Dalal, M., Bahl, S., Lin, S., and Levine, S.
Visual Reinforcement Learning with Imagined Goals.
Proc. of NeurIPS, 2018b.

Nair, A., Bahl, S., Khazatsky, A., Pong, V., Berseth, G., and Levine, S.
Contextual imagined goals for self-supervised robotic learning.
In *Conference on Robot Learning*, pp. 530–539, 2020.

Narayan-Chen, A., Jayannavar, P., and Hockenmaier, J.
Collaborative dialogue in minecraft.
In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 5405–5415, 2019.

Nasiriany, S., Pong, V., Lin, S., and Levine, S.
Planning with goal-conditioned policies.
In *Proc. of NeurIPS*, pp. 14814–14825, 2019.

Ndousse, K. K., Eck, D., Levine, S., and Jaques, N.
Emergent social learning via multi-agent reinforcement learning.
In *International Conference on Machine Learning*. PMLR, 2021.

Ng, A. Y., Russell, S. J., et al.
Algorithms for inverse reinforcement learning.
In *International Conference on Machine Learning*, volume 1, pp. 2, 2000.

Nguyen, K., Misra, D., Schapire, R., Dudík, M., and Shafto, P.
Interactive learning from activity description.
Proc. of ICML, 2021.

Nguyen, M. and Oudeyer, P.-Y.
Socially guided intrinsic motivation for robot learning of motor skills.
Autonomous Robots, 36(3):273–294, 2014.

Nguyen, T. T., Nguyen, N. D., and Nahavandi, S.
Deep reinforcement learning for multiagent systems: A review of challenges, solutions, and applications.
IEEE Transactions on Cybernetics, 50:3826–3839, 2020.

Nguyen-Tuong, D. and Peters, J.
Model Learning for Robot Control: A Survey.
Cognitive processing, 12(4):319–340, 2011.

Oh, J., Singh, S. P., Lee, H., and Kohli, P.
Zero-shot task generalization with multi-task deep reinforcement learning.
In *Proc. of ICML*, volume 70, pp. 2661–2670, 2017.

Oh, J., Guo, Y., Singh, S., and Lee, H.
Self-imitation learning.
In *ICML*, 2018.

Oller, D. K., Griebel, U., Iyer, S. N., Jhang, Y., Warlaumont, A. S., Dale, R., and Call, J.
Language origins viewed in spontaneous and interactive vocal rates of human and bonobo infants.
Frontiers in psychology, 10:729, 2019.

Osa, T., Tangkaratt, V., and Sugiyama, M.
Discovering Diverse Solutions in Deep Reinforcement Learning.
ArXiv - abs/2103.07084, 2021.

Oudeyer, P.-Y.
The self-organization of speech sounds.
Journal of theoretical biology, 233 3:435–49, 2005.

Oudeyer, P.-Y.
Self-organization in the evolution of speech.
In *Oxford Studies in the Evolution of Language*, 2006.

Oudeyer, P.-Y. and Kaplan, F.
What is intrinsic motivation? a typology of computational approaches.
Frontiers in neurorobotics, 1:6, 2007.

Oudeyer, P.-Y. and Kaplan, F.
What Is Intrinsic Motivation? A Typology of Computational Approaches.
Frontiers in neurorobotics, 2009.

Oudeyer, P.-Y. and Smith, L. B.
How evolution may work through curiosity-driven developmental process.
Topics in Cognitive Science, 8(2):492–502, 2016.

Oudeyer, P.-Y., Kaplan, F., and Hafner, V. V.
Intrinsic Motivation Systems for Autonomous Mental Development.
IEEE transactions on evolutionary computation, 11(2):265–286, 2007.

Pathak, D., Agrawal, P., Efros, A. A., and Darrell, T.
Curiosity-driven exploration by self-supervised prediction.
In *Proc. of ICML*, volume 70, pp. 2778–2787, 2017.

Paul, R., Arkin, J., Roy, N., and Howard, T.
Efficient grounding of abstract spatial concepts for natural language interaction with robot manipulators.
In *Robotics: Science and Systems*, 2016.

Perez, E., Strub, F., de Vries, H., Dumoulin, V., and Courville, A.
Film: Visual reasoning with a general conditioning layer, 2017.

Pérolat, J., Leibo, J. Z., Zambaldi, V., Beattie, C., Tuyls, K., and Graepel, T.
A multi-agent reinforcement learning model of common-pool resource appropriation.
In Guyon, I., Luxburg, U. V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems*, volume 30.
Curran Associates, Inc., 2017.
URL https://proceedings.neurips.cc/paper/2017/file/2b0f658cbffd284984fb11d90254081f-Paper.pdf.

Piaget, J.
The Origins of Intelligence in Children.
Translation Margaret Cook – WW Norton & Co, 1952.

Pitis, S., Chan, H., Zhao, S., Stadie, B. C., and Ba, J.
Maximum entropy gain exploration for long horizon multi-goal reinforcement learning.
In *Proc. of ICML*, volume 119, pp. 7750–7761, 2020.

Plappert, M., Andrychowicz, M., Ray, A., McGrew, B., Baker, B., Powell, G., Schneider, J., Tobin, J., Chociej, M., Welinder, P., et al.
Multi-goal reinforcement learning: Challenging robotics environments and request for research.
ArXiv - abs/1802.09464, 2018.

Pomerleau, D. A.
Efficient training of artificial neural networks for autonomous navigation.
Neural computation, 3(1):88–97, 1991.

Pong, V., Dalal, M., Lin, S., Nair, A., Bahl, S., and Levine, S.
Skew-fit: State-covering self-supervised reinforcement learning.
In *Proc. of ICML*, volume 119, pp. 7783–7792, 2020.

Portelance, E., Frank, M. C., Jurafsky, D., Sordoni, A., and Laroche, R.
The emergence of the shape bias results from communicative efficiency.
In *CONLL*, 2021.

Portelas, R., Colas, C., Hofmann, K., and Oudeyer, P.-Y.
Teacher Algorithms for Curriculum Learning of Deep Rl in Continuously Parameterized Environments.
In *Proc. of CoRL*, pp. 835–853, 2020a.

Portelas, R., Colas, C., Weng, L., Hofmann, K., and Oudeyer, P.
Automatic curriculum learning for deep RL: A short survey.
In *Proc. of IJCAI*, pp. 4819–4825, 2020b.

Precup, D.
Temporal abstraction in reinforcement learning.
PhD thesis, The University of Massachussetts, 2000a.

Precup, D.
Temporal Abstraction in Reinforcement Learning.
PhD Thesis, The University of Massachussetts, 2000b.

Puterman, M. L.
Markov decision processes: discrete stochastic dynamic programming.
John Wiley & Sons, 2014.

Racanière, S., Lampinen, A., Santoro, A., Reichert, D., Firoiu, V., and Lillicrap, T.
Automated curricula through setter-solver interactions.
ArXiv - abs/1909.12892, 2019.

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., and Sutskever, I.
Language models are unsupervised multitask learners.
2019.

Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., et al.
Learning Transferable Visual Models from Natural Language Supervision.
Proc. of ICML, 2021.

Raileanu, R. and Rocktäschel, T.
RIDE: rewarding impact-driven exploration for procedurally-generated environments.
In *Proc. of ICLR*, 2020.

Ram, A., Leake, D. B., and Leake, D.
Goal-driven learning.
MIT press, 1995.

Ramesh, A., Pavlov, M., Goh, G., Gray, S., Voss, C., Radford, A., Chen, M., and Sutskever, I.
Zero-Shot Text-to-Image Generation.
ArXiv – abs/2102.12092, 2021.
doi: 10.48550/ARXIV.2102.12092.
URL https://arxiv.org/abs/2102.12092.

Ramesh, A., Dhariwal, P., Nichol, A., Chu, C., and Chen, M.
Hierarchical Text-Conditional Image Generation with CLIP Latents.
ArXiv – abs/2204.06125, 2022.

Ramesh, R., Tomar, M., and Ravindran, B.
Successor options: An option discovery framework for reinforcement learning.
In *Proc. of IJCAI*, pp. 3304–3310, 2019.

Reed, S., Zolna, K., Parisotto, E., Colmenarejo, S. G., Novikov, A., Barth-maron, G., Giménez, M., Sulsky, Y., Kay, J., Springenberg, J. T., Eccles, T., Bruce, J., Razavi, A., Edwards, A., Heess, N., Chen, Y., Hadsell, R., Vinyals, O., Bordbar, M., and de Freitas, N.
A generalist agent.
Transactions on Machine Learning Research, 2022.
URL https://openreview.net/forum?id=1ikK0kHjvj.
Featured Certification.

Ren, Y., Guo, S., Labeau, M., Cohen, S. B., and Kirby, S.
Compositional languages emerge in a neural iterated learning model.
In *International Conference on Learning Representations*, 2020.
URL https://openreview.net/forum?id=HkePNpVKPB.

Röder, F., Eppe, M., Nguyen, P. D., and Wermter, S.
Curious hierarchical actor-critic reinforcement learning.
In *International Conference on Artificial Neural Networks*, pp. 408–419. Springer, 2020.

Rodríguez Luna, D., Ponti, E. M., Hupkes, D., and Bruni, E.
Internal and external pressures on language emergence: least effort, object constancy and frequency.
In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pp. 4428–4437, Online, November 2020. Association for Computational Linguistics.
doi: 10.18653/v1/2020.findings-emnlp.397.
URL https://aclanthology.org/2020.findings-emnlp.397.

Rolf, M. and Steil, J. J.
Efficient exploratory learning of inverse kinematics on a bionic elephant trunk.
IEEE transactions on neural networks and learning systems, 25(6):1147–1160, 2013.

Rolf, M., Steil, J. J., and Gienger, M.
Goal babbling permits direct learning of inverse kinematics.
IEEE Transactions on Autonomous Mental Development, 2(3):216–229, 2010.

Rombach, R., Blattmann, A., Lorenz, D., Esser, P., and Ommer, B.
High-resolution image synthesis with latent diffusion models.
ArXiv, abs/2112.10752, 2021.

Roy, J., Barde, P., Harvey, F., Nowrouzezahrai, D., and Pal, C.
Promoting coordination through policy regularization in multi-agent deep reinforcement learning.
In *Advances in Neural Information Processing Systems*, volume 33, pp. 15774–15785, 2020.

Ruis, L., Andreas, J., Baroni, M., Bouchacourt, D., and Lake, B. M.
A benchmark for systematic generalization in grounded language understanding.
In *Proc. of NeurIPS*, 2020.

Rumelhart, D. E., Smolensky, P., McClelland, J. L., and Hinton, G.
Sequential Thought Processes in Pdp Models.
Parallel distributed processing: explorations in the microstructures of cognition, 2:3–57, 1986.

Runco, M. A. and Jaeger, G. J.
The Standard Definition of Creativity.
Creativity Research Journal, (1), 2012.
ISSN 1040-0419, 1532-6934.

Saharia, C., Chan, W., Saxena, S., Li, L., Whang, J., Denton, E. L., Ghasemipour, S. K. S., Ayan, B. K., Mahdavi, S. S., Lopes, R. G., Salimans, T., Ho, J., Fleet, D., and Norouzi, M.
Photorealistic text-to-image diffusion models with deep language understanding.
ArXiv, abs/2205.11487, 2022.

Salimans, T., Ho, J., Chen, X., Sidor, S., and Sutskever, I.
Evolution strategies as a scalable alternative to reinforcement learning.
ArXiv - abs/1703.03864, 2017.

Santoro, A., Raposo, D., Barrett, D. G. T., Malinowski, M., Pascanu, R., Battaglia, P., and Lillicrap, T.
A simple neural network module for relational reasoning, 2017.

Santoro, A., Lampinen, A., Mathewson, K., Lillicrap, T., and Raposo, D.
Symbolic behaviour in artificial intelligence.
ArXiv – abs/2102.03406, 2021.

Santucci, V. G., Baldassarre, G., and Mirolli, M.
Grail: a goal-discovering robotic architecture for intrinsically-motivated learning.
IEEE Transactions on Cognitive and Developmental Systems, 8(3):214–231, 2016.

Santucci, V. G., Oudeyer, P.-Y., Barto, A., and Baldassarre, G.
Intrinsically motivated open-ended learning in autonomous robots.
Frontiers in Neurorobotics, 13:115, 2020.

Schaal, S.
Dynamic movement primitives -a framework for motor control in humans and humanoid robotics.
2006.

Schaul, T., Horgan, D., Gregor, K., and Silver, D.
Universal value function approximators.
In *Proc. of ICML*, volume 37, pp. 1312–1320, 2015a.

Schaul, T., Horgan, D., Gregor, K., and Silver, D.
Universal Value Function Approximators.
Proc. of ICML, 2015b.

Schlegel, M., Jacobsen, A., Abbas, Z., Patterson, A., White, A., and White, M.
General value function networks.
Journal of Artificial Intelligence Research, 70:497–543, 2021.

Schmidhuber, J.
Making the World Differentiable: On Using Self-Supervised Fully Recurrent N eu al Networks for Dynamic Reinforcement Learning and Planning in Non-Stationary Environments, 1990.

Schmidhuber, J.
Curious model-building control systems.
In *Neural Networks, 1991. 1991 IEEE International Joint Conference on*, pp. 1458–1463. IEEE, 1991a.

Schmidhuber, J.
Learning to generate sub-goals for action sequences.
In *Artificial neural networks*, pp. 967–972, 1991b.

Schmidhuber, J.
A possibility for implementing curiosity and boredom in model-building neural controllers.
In *Proc. of the international conference on simulation of adaptive behavior: From animals to animats*, pp. 222–227, 1991c.

Schmidhuber, J.
Curious Model-Building Control Systems.
IEEE International Joint Conference on Neural Networks, 1991d.

Schmidhuber, J.
Formal theory of creativity, fun, and intrinsic motivation (1990–2010).
IEEE Transactions on Autonomous Mental Development, 2(3):230–247, 2010.

Schramowski, P., Turan, C., Andersen, N., Rothkopf, C. A., and Kersting, K.
Large Pre-Trained Language Models Contain Human-Like Biases of What Is Right and Wrong to Do.
Nature Machine Intelligence, (3), 2022.

Schrittwieser, J., Antonoglou, I., Hubert, T., Simonyan, K., Sifre, L., Schmitt, S., Guez, A., Lockhart, E., Hassabis, D., Graepel, T., Lillicrap, T., and Silver, D.
Mastering Atari, Go, Chess and Shogi by Planning with a Learned Model.
Nature, 588(7839):604–609, 2020.
ISSN 0028-0836, 1476-4687.

Sehnke, F., Osendorfer, C., Rückstieß, T., Graves, A., Peters, J., and Schmidhuber, J.
Parameter-exploring policy gradients.
Neural Networks, 23(4):551–559, 2010.

Sekar, R., Rybkin, O., Daniilidis, K., Abbeel, P., Hafner, D., and Pathak, D.
Planning to explore via self-supervised world models.
In *Proc. of ICML*, volume 119, pp. 8583–8592, 2020.

Shah, D. S., Schwartz, H. A., and Hovy, D.
Predictive biases in natural language processing models: A conceptual framework and overview.
In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 5248–5264, Online, 2020. Association for Computational Linguistics.

Shanahan, M. and Mitchell, M.
Abstraction for Deep Reinforcement Learning.
Proc. of IJCAI, 2022.

Sharma, A., Gu, S., Levine, S., Kumar, V., and Hausman, K.
Dynamics-aware unsupervised discovery of skills.
In *Proc. of ICLR*, 2020.

Sharma, P., Torralba, A., and Andreas, J.
Skill Induction and Planning with Latent Language.
Proc. of ACL, 2021.

Shridhar, M., Yuan, X., Cote, M.-A., Bisk, Y., Trischler, A., and Hausknecht, M.
ALFWorld: Aligning Text and Embodied Environments for Interactive Learning.
Proc. of ICLR, 2021.

Sigaud, O., Caselles-Dupré, H., Colas, C., Akakzia, A., Oudeyer, P.-Y., and Chetouani, M.
Towards teachable autonomous agents.
ArXiv - abs/2105.11977, 2021a.

Sigaud, O., Colas, C., Akakzia, A., Chetouani, M., and Oudeyer, P.-Y.
Towards Teachable Autonomous Agents.
ArXiv – abs/2105.11977, 2021b.

Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., Van Den Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M., et al.
Mastering the game of go with deep neural networks and tree search.
In *nature*, volume 529, pp. 484–489. Nature Publishing Group, 2016.

Simsek, Ö. and Barto, A. G.
Using relative novelty to identify useful temporal abstractions in reinforcement learning.
In *Proc. of ICML*, volume 69, 2004.

Simsek, Ö. and Barto, A. G.
Skill characterization based on betweenness.
In *Proc. of NeurIPS*, pp. 1497–1504, 2008.

Singh, S., Lewis, R. L., Barto, A. G., and Sorg, J.
Intrinsically motivated reinforcement learning: An evolutionary perspective.
IEEE Transactions on Autonomous Mental Development, 2(2):70–82, 2010.

Spelke, E. S., Breinlinger, K., Macomber, J., and Jacobson, K.
Origins of knowledge.
Psychological review, 99(4):605, 1992.

Sperber, D., Premack, D., and Premack, A. J.
Causal Cognition: A Multidisciplinary Debate.
Clarendon Press Oxford, 1995.

Stanley, K. O.
Why Open-Endedness Matters.
Artificial life, 25(3):232–235, 2019.

Stanley, K. O. and Soros, L.
The Role of Subjectivity in the Evaluation of Open-Endedness.
In *Presentation delivered in OEE2: The Second Workshop on Open-Ended Evolution, at ALIFE 2016*, 2016.

Steels, L.
A self-organizing spatial vocabulary.
Artificial life, 2(3):319–332, 1995a.

Steels, L.
Semiotic dynamics for embodied agents.
IEEE Intelligent Systems, 21(3):32–38, 2006.
doi: 10.1109/MIS.2006.58.

Steels, L. L.
A self-organizing spatial vocabulary.
Artificial Life, 2:319–332, 1995b.

Steels, L. L.
Language games for autonomous robots.
IEEE Intelligent Systems, 16:16–22, 2001.

Steels, L. L.
The Talking Heads experiment.
Number 1 in Computational Models of Language Evolution. Language Science Press, Berlin, 2015.
doi: 10.17169/FUDOCS_document_000000022455.

Steels, L. L. and Loetzsch, M.
The grounded naming game.
2012.

Stevens, K. N.
On the quantal nature of speech.
Journal of Phonetics, 17(1):3–45, 1989.
ISSN 0095-4470.
doi: https://doi.org/10.1016/S0095-4470(19)31520-7.
URL https://www.sciencedirect.com/science/article/pii/S0095447019315207.

Stooke, A., Mahajan, A., Barros, C., Deck, C., Bauer, J., Sygnowski, J., Trebacz, M., Jaderberg, M., Mathieu, M., et al.
Open-ended learning leads to generally capable agents.
ArXiv - abs/2107.12808, 2021.

Sugita, Y. and Tani, J.
Learning semantic combinatoriality from the interaction between linguistic and behavioral processes.
Adaptive behavior, 13(1):33–52, 2005.

Sukhbaatar, S., Lin, Z., Kostrikov, I., Synnaeve, G., Szlam, A., and Fergus, R.
Intrinsic motivation and automatic curricula via asymmetric self-play.
In *Proc. of ICLR*, 2018.

Sutton, R. S. and Barto, A. G.
Introduction to Reinforcement Learning.
MIT press Cambridge, 1998.

Sutton, R. S. and Barto, A. G.
Reinforcement learning: An introduction.
MIT press, 2018.

Sutton, R. S. and Tanner, B.
Temporal-difference networks.
In Saul, L., Weiss, Y., and Bottou, L. (eds.), *Advances in Neural Information Processing Systems*, volume 17. MIT Press, 2004.
URL https://proceedings.neurips.cc/paper/2004/file/9d28de8ff9bb6a3fa41fddfdc28f3bc1-Paper.pdf.

Sutton, R. S., Precup, D., and Singh, S. P.
Intra-option learning about temporally abstract actions.
In *Proc. of ICML*, volume 98, pp. 556–564, 1998.

Sutton, R. S., Precup, D., and Singh, S.
Between MDPs and Semi-MDPs: A Framework for Temporal Abstraction in Reinforcement Learning.
Artificial intelligence, (1-2), 1999.
Publisher: Elsevier.

Sutton, R. S., Modayil, J., Delp, M., Degris, T., Pilarski, P. M., White, A., and Precup, D.
Horde: A scalable real-time architecture for learning knowledge from unsupervised sensorimotor interaction.
In *The 10th International Conference on Autonomous Agents and Multiagent Systems-Volume 2*, pp. 761–768, 2011.

Tam, A. C., Rabinowitz, N. C., Lampinen, A. K., Roy, N. A., Chan, S. C. Y., Strouse, D., Wang, J. X., Banino, A., and Hill, F.
Semantic Exploration from Language Abstractions and Pretrained Representations.
ArXiv – abs/2204.05080, 2022.

Tani, J.
Exploring robotic minds: actions, symbols, and consciousness as self-organizing dynamic phenomena.
Oxford University Press, 2016.

Taniguchi, T., Nagai, T., Nakamura, T., Iwahashi, N., Ogata, T., and Asoh, H.
Symbol emergence in robotics: a survey.
Advanced Robotics, 30(11-12):706–728, 2016.

Tasse, G. N., James, S. D., and Rosman, B.
A boolean task algebra for reinforcement learning.
In *Proc. of NeurIPS*, 2020.

Tomasello, M.
The cultural origins of human cognition.
Harvard University Press, 1999a.
ISBN 9780674005822.

Tomasello, M.
The Cultural Origins of Human Cognition.
Harvard University Press, 1999b.
ISBN 978-0-674-00582-2.

Tomasello, M.
The item-based nature of children's early syntactic development.
Trends in cognitive sciences, 4(4):156–163, 2000.

Tomasello, M.
Constructing a Language.
Harvard university press, 2009.

Tomasello, M.
Becoming Human – A Theory of Ontogeny.
Harvard University Press, Cambridge, MA and London, England, 2019.
ISBN 9780674988651.
doi: doi:10.4159/9780674988651.
URL https://doi.org/10.4159/9780674988651.

Tomasello, M. and Olguin, R.
Twenty-three-month-old children have a grammatical category of noun.
Cognitive development, 8(4):451–464, 1993.

Tomasello, M., Carpenter, M., Call, J., Behne, T., and Moll, H.
Understanding and Sharing Intentions: The Origins of Cultural Cognition.
Behavioral and brain sciences, 2005.

Tuci, E., Ferrauto, T., Zeschel, A., Massera, G., and Nolfi, S.
An experiment on behavior generalization and the emergence of linguistic compositionality in evolving robots.
IEEE Transactions on Autonomous Mental Development, 3(2):176–189, 2011.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I.
Attention is all you need.
In *Proc. of NeurIPS*, pp. 5998–6008, 2017.

Veeriah, V., Oh, J., and Singh, S.
Many-goals reinforcement learning.
ArXiv - abs/1806.09605, 2018.

Venkattaramanujam, S., Crawford, E., Doan, T., and Precup, D.
Self-supervised learning of distance functions for goal-conditioned reinforcement learning.
2019.

Vezhnevets, A. S., Osindero, S., Schaul, T., Heess, N., Jaderberg, M., Silver, D., and Kavukcuoglu, K.
Feudal networks for hierarchical reinforcement learning.
In *Proc. of ICML*, volume 70, pp. 3540–3549, 2017.

Vollmer, A.-L., Grizou, J., Lopes, M., Rohlfing, K., and Oudeyer, P.-Y.
Studying the co-construction of interaction protocols in collaborative tasks with humans.
In *4th International Conference on Development and Learning and on Epigenetic Robotics*, pp. 208–215. IEEE, 2014.

Vollmer, A.-L., Wrede, B., Rohlfing, K. J., and Oudeyer, P.-Y.
Pragmatic frames for teaching and learning in human–robot interaction: Review and challenges.
Frontiers in neurorobotics, 10:10, 2016.

Vygotsky, L. S.
Thought and Language.
MIT press, 1934.

Vygotsky, L. S.
Tool and Symbol in Child Development.
In *Mind in Society*, chapter Tool and Symbol in Child Development, pp. 19–30. Harvard University Press, 1978.
ISBN 0674576292.

Vyshedskiy, A.
Language Evolution to Revolution: the Leap From Rich-Vocabulary Non-Recursive Communication System to Recursive Language 70,000 Years Ago Was Associated with Acquisition of a Novel Component of Imagination, Called Prefrontal Synthesis, Enabled By a Mutation that Slowed Down the Prefrontal Cortex Maturation Simultaneously in Two or More Children – the Romulus and Remus Hypothesis.
Research Ideas and Outcomes, 2019.
ISSN 2367-7163.

Warde-Farley, D., de Wiele, T. V., Kulkarni, T. D., Ionescu, C., Hansen, S., and Mnih, V.
Unsupervised Control Through Non-Parametric Discriminative Rewards.
In *Proc. of ICLR*, 2019.

Waxman, S. R. and Markow, D. B.
Words as Invitations to Form Categories: Evidence from 12-to 13-Month-Old Infants.
Cognitive psychology, (3), 1995.

Weidinger, L., Mellor, J., Rauh, M., Griffin, C., Uesato, J., Huang, P.-S., Cheng, M., Glaese, M., Balle, B., Kasirzadeh, A., Kenton, Z., Brown, S., Hawkins, W., Stepleton, T., Biles, C., Birhane, A., Haas, J., Rimell, L., Hendricks, L. A., Isaac, W., Legassick, S., Irving, G., and Gabriel, I.
Ethical and social risks of harm from language models.
ArXiv – abs/2112.04359, 2021.

West, P., Bhagavatula, C., Hessel, J., Hwang, J. D., Jiang, L., Bras, R. L., Lu, X., Welleck, S., and Choi, Y.
Symbolic Knowledge Distillation: from General Language Models to Commonsense Models.
Proc. of NAACL, 2022.

Whorf, B. L.
Language, Thought, and Reality: Selected Writings of Benjamin Lee Whorf.
MIT press, 1956.

Wierstra, D., Schaul, T., Glasmachers, T., Sun, Y., Peters, J., and Schmidhuber, J.
Natural evolution strategies.
The Journal of Machine Learning Research, 15(1):949–980, 2014.

Wittgenstein, L.
Philosophical Investigations.
John Wiley & Sons, 1953.

Witty, S., Lee, J. K., Tosch, E., Atrey, A., Clary, K., Littman, M. L., and Jensen, D.
Measuring and Characterizing Generalization in Deep Reinforcement Learning.
Applied AI Letters, 2021.

Wong, C., Ellis, K., Tenenbaum, J. B., and Andreas, J.
Leveraging Language to Learn Program Abstractions and Search Heuristics.
Proc. of ICML, 2021.

Wood, D., Bruner, J. S., and Ross, G.
The Role of Tutoring in Problem Solving.
Journal of Child Psychology and Psychiatry, 17(2), 1976.
ISSN 0021-9630, 1469-7610.

Woodward, M., Finn, C., and Hausman, K.
Learning to interactively learn and assist.
In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pp. 2535–2543, 2020.

Wu, Y., Tucker, G., and Nachum, O.
The laplacian in RL: learning representations with efficient approximations.
In *Proc. of ICLR*, 2019.

Xie, T., Langford, J., Mineiro, P., and Momennejad, I.
Interaction-grounded learning.
In Meila, M. and Zhang, T. (eds.), *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 11414–11423. PMLR, 18–24 Jul 2021.
URL https://proceedings.mlr.press/v139/xie21e.html.

Yan, C., Carnevale, F., Georgiev, P., Santoro, A., Guy, A., Muldal, A., Hung, C.-C., Abramson, J., Lillicrap, T., and Wayne, G.
Intra-agent speech permits zero-shot task acquisition.
ArXiv – abs/2206.03139, 2022.

Yi, K., Gan, C., Li, Y., Kohli, P., Wu, J., Torralba, A., and Tenenbaum, J. B.
Clevrer: Collision events for video representation and reasoning, 2020.

Yoshida, H. and Smith, L. B.
Sound Symbolism and Early Word Learning in Two Languages.
In *Proceedings of the Annual Meeting of the Cognitive Science Society*, 2003.

Yuan, X., Côté, M.-A., Fu, J., Lin, Z., Pal, C., Bengio, Y., and Trischler, A.
Interactive language learning by question answering.
In *Proc. of EMNLP*, pp. 2796–2813. Association for Computational Linguistics, 2019b.

Yuan, X., Côté, M.-A., Fu, J., Lin, Z., Pal, C., Bengio, Y., and Trischler, A.
Interactive Language Learning by Question Answering.
In *Proc. of EMNLP*. Association for Computational Linguistics, 2019a.

Zaheer, M., Kottur, S., Ravanbakhsh, S., Póczos, B., Salakhutdinov, R., and Smola, A. J.
Deep sets.
In *Proc. of NeurIPS*, pp. 3391–3401, 2017.

Zaremba, W., Sutskever, I., and Vinyals, O.
Recurrent neural network regularization.
ArXiv - abs/1409.2329, 2014.

Zeng, A., Wong, A., Welker, S., Choromanski, K., Tombari, F., Purohit, A., Ryoo, M., Sindhwani, V., Lee, J., Vanhoucke, V., et al.
Socratic models: Composing zero-shot multimodal reasoning with language.
ArXiv – abs/2204.00598, 2022.

Zhang, Y., Abbeel, P., and Pinto, L.
Automatic curriculum learning through value disagreement.
In *Proc. of NeurIPS*, 2020.

Zhou, H., Kadav, A., Lai, F., Niculescu-Mizil, A., Min, M. R., Kapadia, M., and Graf, H. P.
Hopper: Multi-hop transformer for spatiotemporal reasoning.
In *International Conference on Learning Representations*, 2021.
URL https://openreview.net/forum?id=MaZFq7bJif7.

Zhou, L. and Small, K.
Inverse reinforcement learning with natural language goals, 2020a.

Zhou, L. and Small, K.
Inverse Reinforcement Learning with Natural Language Goals.
ArXiv – abs/2008.06924, 2020b.

Zhu, Y., Mottaghi, R., Kolve, E., Lim, J. J., Gupta, A., Fei-Fei, L., and Farhadi, A.
Target-driven visual navigation in indoor scenes using deep reinforcement learning.
In *2017 IEEE international conference on robotics and automation (ICRA)*, pp. 3357–3364. IEEE, 2017.

Zlatev, J.
The Epigenesis of Meaning in Human Beings, and Possibly in Robots.
Minds and Machines, (2), 2001.
ISSN 1572-8641.

Zuidema, W. and De Boer, B.
The evolution of combinatorial phonology.
Journal of Phonetics, 37(2):125–144, 2009.

Zwaan, R. and Madden, C.
Embodied sentence comprehension.
Grounding Cognition: The Role of Perception and Action in Memory, Language, and Thinking, pp. 224–245, 2005a.
doi: 10.1017/CBO9780511499968.010.

Zwaan, R. A. and Madden, C. J.
Embodied Sentence Comprehension, pp. 224–245.
Cambridge University Press, 2005b.
doi: 10.1017/CBO9780511499968.010.

List of Figures

List of Tables