Towards Social Auotelic Artificial Agents

Formation and Exploitation of Cultural Conventions in Autonomous Embodied 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 March 10, 2023. Defended on May 8, 2023.

Composition of the jury:

Pr. Noah GOODMAN	Associate Professor	Stanford University	Reviewer
Pr. Bruno GALANTUCCI	Full Professor	Yeshiva University	Reviewer
Pr. Stefano PALMINTERI	Research Director	ENS	Examinator
Dr. Andrew LAMPINEN	Senior Research Scientist	Deepmind	Examinator
Dr. Xavier HINAUT	Researcher	INRIA	Examinator
Dr. Pierre-Yves OUDEYER	Research Director	INRIA	Director
Dr. Clément MOULIN-FRIER	Researcher	INRIA	Director

Contents

Acknowledgments

Abstract

Glossary

Action-value Function The action-value function $Q_\pi(s, a)$ is the expected return of the trajectory taking action a from state s before following π from the next state s'. 12

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 Convention any social production, linguistic or physical, internal or interpersonal, used to communicate, cooperate, teach, think, or transmit.. 5

Developmental Artificial Intelligence a multidisciplinary field that integrates principles from artificial intelligence, developmental psychology, and neuroscience to simulate and analyze the cognitive mechanisms of artificial agents. 24

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.. 18

Goal-achievement function $R_g(\cdot) = R_\mathcal{G}(\cdot \mid z_g)$ where $R_\mathcal{G}$ is a goal-conditioned reward function.. 18

Goal-conditioned policy a function that generates the next action given the current state and the goal.. 18

Markov Decision Process A Markov Decision Process (MDP) models a decision-making problem using a set of states, a set of actions, and a set of probabilities that describe the outcome of each action in each state.. 10

Markov Game The framework of Markov Games is a multi-agent extension of MDPs.. 21

Self-organization process by which spontaneously ordered patterns and structures emerge in a system without the need for central control or external guidance. 25

Skill the association of a goal and a policy to reach it.. 18

Value Function The value function $V_\pi(s)$ of a policy π gives the expected return of a trajectory starting from s and following π. 12

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 Reinforcement Learning (MARL) (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, 1999; 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 Fig. 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 Autotelic Artificial Intelligence* which enables sociocultural interactions to transform agents' learning signal, yielding better learners.

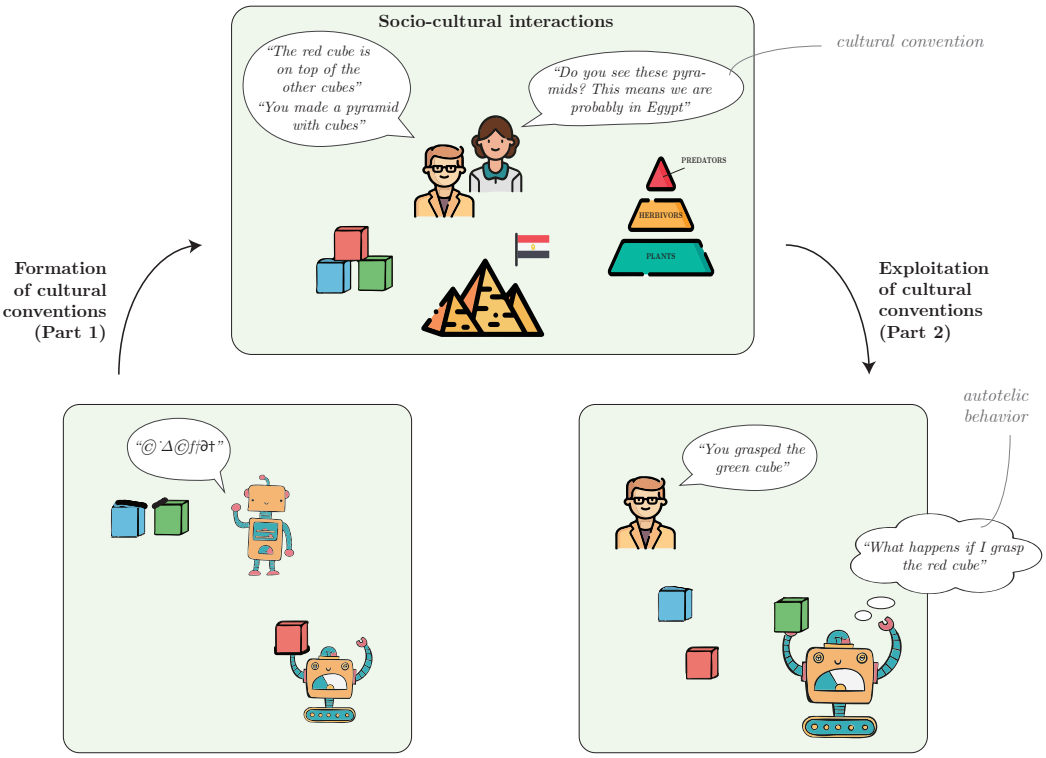


Figure 1.1: **Dual organization of the present research.** In the first part we take a bottom-up approach and study the self-organization of cultural conventions in artificial agents from social interactions. In the second part, we use a top-down approach to investigate the impact of pre-existing cultural conventions on artificial agents when they interact with social peers.

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. We then close it with a short intuitive explanation of the position of this research with respect to other paradigms in AI and a summary of our contributions.

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, 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?

1.1.1 Humans are autotelic learners

A central aspect of human development is the notion of goal. Studying the use of the notion of goal in past psychological research, Elliot & Fryer (2008) propose the following general definition:

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

A goal is therefore a future projection that influences human behaviors. During exploratory play, children constantly invent and pursue their own problems/goals (Chu & Schulz, 2020). In particular, children's exploration seems to be driven by intrinsically motivated brain processes that trigger spontaneous exploration for the mere purpose of visiting interesting situations (Gopnik et al., 1999; Kaplan & Oudeyer, 2007; Kidd & Hayden, 2015b). But how do we measure interestingness? Hunt (1965) propose to evaluate situations in term of *optimal incongruity*. Similarly, Berlyne (1966) suggest relying on the notion of *intermediate level of novelty* while Kidd et al. (2012) showed that young infants focus on goals with *intermediate complexity*. Finally, Csikzentmihalyi (1997), in his flow theory suggests that for human beings to feel pleasure during learning they should target goals with *optimal challenge*. He uses the term *autotelic* to describe intrinsically motivated agents that are in the flow state.

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*.

1.1.2 Humans are social learners

Social interactions are another crucial property of human development. At birth, humans enter a culture that strongly shapes their development (Whorf, 1956). Humans are social beings; intrinsically motivated to interact and cooperate with their peers (Tomasello, 1999; Tomasello et al., 2005; Brewer et al., 2014). Indeed, we use social interactions and language at every stage of our development to communicate, cooperate, teach and organize our thoughts.

Cooperation

First, social interactions enable us to **cooperate**, to jointly commit to shared goals. Tomasello (2019) describes this collaborative behavior as *shared intentionality*. According to him, shared intentionality arises around nine months and enables us to relate to others as equals and to align on low-level common goals such as "looking in the same direction". Shared intentionality allows us to mentally represent and then adopt another's goal. It is thus very linked to the theory of mind (Wellman, 1992). It allows us to share goals, emotions, attention, or even knowledge. As we grow older, shared intentionality becomes *collective intentionality* and allows us to be part of a society in which goals are associated with social norms and conventions. In a recent study, Mcclung et al. (2017) use an egg hunt game to show that group membership and the ability to talk led to increased collaboration between participants. By analyzing the conversation they found that in-group participants were talking about the hunt in terms of a shared or common goal, while out-group participants used individual goals.

Teaching

In a more structured way, social interactions also enable us to **teach**. The idea that social interactions provide a structure for teaching has been supported by many researchers including Vygotsky (1933); Bruner (1985); Rohlfing et al. (2016); Vollmer et al. (2016). Bruner (1985) specifically proposed the concept of *pragmatic frames*: patterns of behaviors that are used to achieve a goal and that are developed through repeated and sequential interactions between a teacher and a learner. According to Bruner, pragmatic frames are made of two key components: 1) a *syntax* which is the observable part of the interactions and includes the sensory means (modalities) as well as the role of each actor; 2) a *meaning* which is the learning content. In his book, Bruner (1985) takes the example of the *book-reading frame* during which the teacher points and asks for labels before providing feedback and correcting the learner depending on their answer. In this case, the pointing/asking/answering mechanism is the syntax and the label is the meaning. Pragmatic frames can happen in a variety of modalities but as we just saw with the book-reading frame, they are often multi-modal and imply linguistic interactions.

Pragmatic frames may also adapt to the learners' abilities. In Vygotsky's *zone of proximal development* (Vygotsky, 1934), 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. In this zone, children can benefit from these social interactions to achieve more than they could alone as illustrated in Fig. 1.2. In Vygotsky's terms, the ZPD is defined as

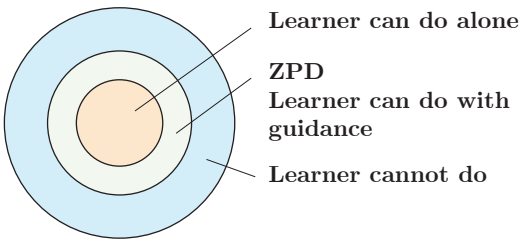


Figure 1.2: ZPD Illustration

> "*the distance between the actual developmental level as determined by independent problem solving and the level of potential development as determined through problem-solving under adult guidance, or in collaboration with more capable peers*" (Vygotsky, 1934)

Thoughts

The language we use in social interaction can also be a cognitive tool that facilitates **thinking**. In Vygotsky's theory, children *internalize* linguistic and social aids and progressively turn these interpersonal processes into intrapersonal *psychological tools*. 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 inner speech would enable *thinking in language* (Carruthers, 1998). The relation between language and thought in humans is the subject of a great debate and will be discussed in greater detailed in chapter **??**, section **??** when introducing the Vyogtskian Auotelic AI framework.

Cultural ratchet

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 generations (Henrich & McElreath, 2003; Morgan et al., 2015; Chopra et al., 2019) — a process known as the *cultural rachet* (Tomasello, 1999). 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, and what to expect from others (Bruner, 1990)

Cultural Convention

In light of the various properties of social interactions presented in this section, we introduce the notion of *cultural convention* which generalizes pragmatic frames to internal (intrapersonal) social production. More specifically, we propose the following definition.

Definition

Cultural Convention: Any social production, linguistic or physical, internal or interpersonal, used to communicate, cooperate, teach, think, or transmit.

1.2 Towards Interactive Social Autonomous Agents

The present research aims at bridging developmental psychology with recent AI methods used to design embodied artificial agents. Building on the "autotelic" and the "cultural convention" notions, our goal is to build interactive social autotelic agents. For this purpose, we immerse artificial agents in social contexts and equip them with learning mechanisms to either construct cultural conventions (in part I) or to exploit cultural conventions to discover new skilss (in part II).

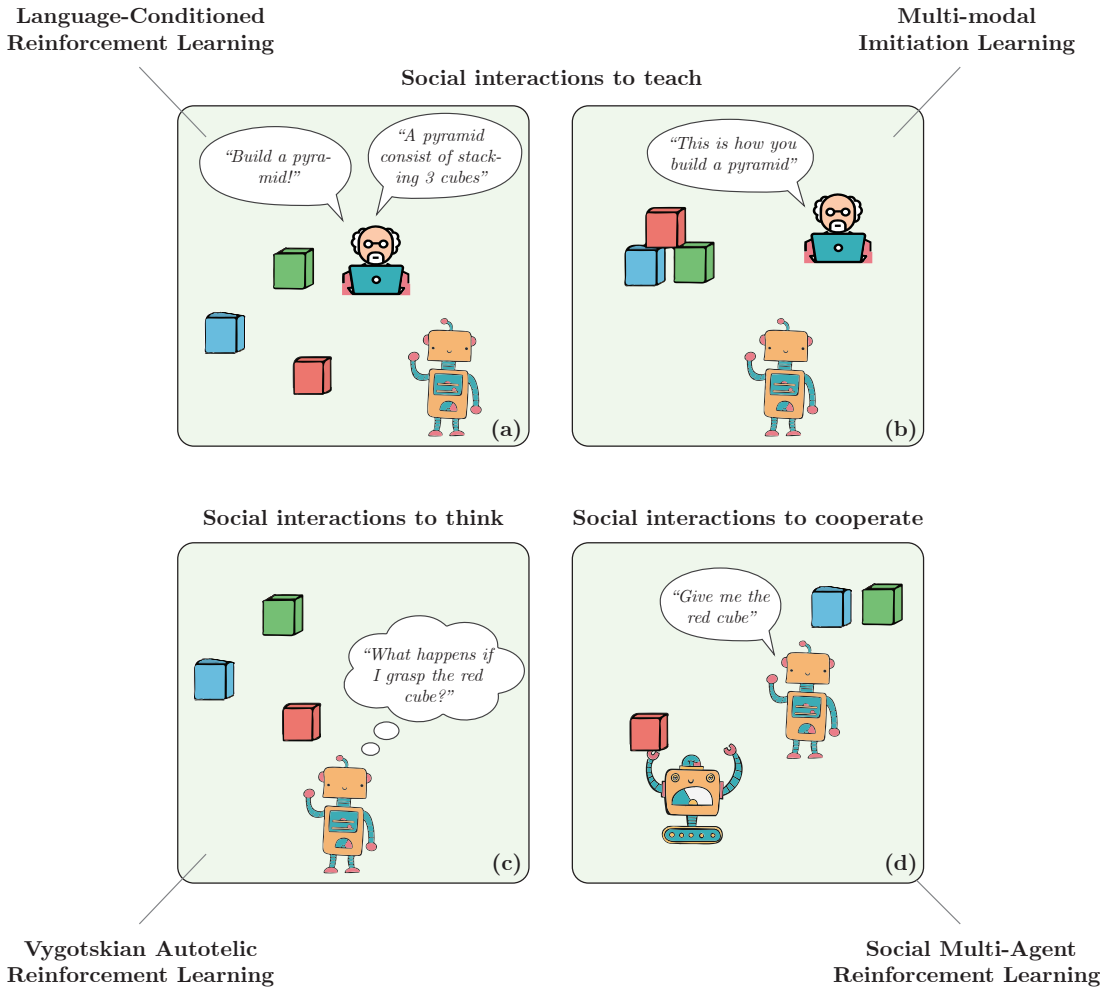


Figure 1.3: **Social Interactions in different ai paradigms.** Social interactions and language instructions are used in both RL and IL setting to guide learners. Language can also serve as a cognitive tool to represent goals in autotelic learning. Finally, they can help agents communicate and cooperate in MARL.

The immersion of artificial agents in social worlds does not require starting from fresh grounds. In fact, numerous works already include social elements in pre-existing AI paradigms. In a recent survey, Luketina et al. (2019) review several approaches instructing RL agents with language, either to condition them or to assist them as displayed in Fig. 1.3 (a). Similarly, recent IL settings have had their training datasets augmented with linguistic descriptions of expert trajectories (Shridhar et al., 2020; Pashevich et al., 2021) as displayed

in Fig. 1.3 (b). In the present research, we will demonstrate that agents can use language as a cognitive tool to imagine creative goals (Colas et al., 2022a) as illustrated in Fig. 1.3 (c). Finally, Jaques et al. (2019) recently presented a MARL framework where agents use social motivations to solve collaborative tasks such as the one depicted in Fig. 1.3(d).

Objectives

The general objective of the present research is to investigate the two following questions:

- **How can cultural conventions self-organize when artificial agents interact?** The objective of the first part of this research is to investigate the key mechanisms required for the **formation** of cultural conventions between artificial agents. In part I, we place ourselves in a multi-agent setup and consider social interactions between two artificial agents that both integrate learning dynamics.
- **How can artificial agents benefit from pre-existing cultural conventions?** Conversely, part II aims at exploring the **exploitation** of pre-existing cultural conventions by autonomous agents. As such, we will consider a single artificial agent interacting with a simulated social partner.

Contributions

The present manuscript starts with an overview of foundational AI paradigms, namely RL, IL, and MARL (chapter 2). Following this, we present the two primary research questions we address here, which are organized around the theme of self-organization: 1) self-organization of cultural convention, and 2) self-organization of trajectories derived from existing cultural conventions.

Our first experimental contribution (chapter 4) investigates the role of sensorimotor constraints in the formation of a graphical language. For this experiment, we place ourselves in the context of Language Games (Steels, 2001) and consider speaker and listener agents exchanging utterances to refer to visual objects. In our setup, utterances are graphical signs produced by a robotic arm and objects are combinations of MNIST digits. We propose a new multi-modal contrastive learning algorithm to enable agents to self-organize a shared communication system in such a sensorimotor setting.

Our second experinental contribution (chapter 5) studies the collaboration between two artificial agents in the *Architect-Builder Problem*: a new interactive setting in which agents have asymmetrical roles and must cooperate to build structures. More specifically, the architect knows the structure that needs to be assembled but cannot act on the blocks of the environment while the builder does not know the task at hand but can manipulate the objects. Our proposed algorithmic solution builds on the shared intentionality and pragmatic frame concepts to enable the architect and the builder to agree on a cultural convention enabling them to solve the task.

Our next contribution (chapter **??**) introduces the Vyogtksian Autotelic AI framework (VAAI). Inspired by the pioneering work of the developmental psychologist Vygotsky (1934), we draw the contour of a more human-like AI where agents are immersed in rich socio-cultural

worlds. By exposing agents to our culture, and enabling them to internalize pre-existing cultural conventions they can use language as a cognitive tool to become better learners.

The VAAI framework is the foundation of two other experimental contributions. Our fourth contribution (chapter **??**) explores how embodied artificial agents can align their trajectories with linguistic descriptions provided by a social partner. This alignment is known as the Language Grounding Problem (Glenberg & Kaschak, 2002; Zwaan & Madden, 2005). We consider the grounding of descriptions involving spatio-temporal concepts and study the impact of architectural biases by testing different variants of multi-modal transformers.

Finally, in our fifth and last contribution (chapter **??**), we implement an autotelic agent that converts linguistic descriptions given by a social partner into targetable goals. We coined this agent IMAGINE. IMAGINE operates in two phases. First, the agent learns to represent, detect and achieve goals by interacting with a social partner. Once it has discovered a variety of interesting interactions doing so, IMAGINE then switches to an autonomous phase and uses language as a cognitive tool to imagine new goal constructs leveraging language compositionality. We show that this algorithm enables agents to discover a greater variety of skills paving the way to more open-ended learning learners.

1.2.1 Collaborations

The present research is the result of multiple collaborations involving several research institutions including INRIA in France, Mila in Canada, and Microsoft Research at Cambridge (UK). My two amazing supervisors, Clément Moulin-Frier and Pierre-Yves Oudeyer from the Flowers Lab (INRIA) were involved in all these collaborations. Our first contribution (chapter 4) was developed during the brilliant internship of Yoann Lemesle (Paris-Dauphine-PSL University) which I had the chance to supervise. Our second contribution (chapter 5) was led by the great Paul Barde (Mila) and myself, under the joint supervision of my and Paul Barde's supervisors, namely Derek Nowrouzezahrai (McGill University) and Chris Pal (Polytechnique Montreal & Mila, CIFAR AI Chair). Most of the work on Vygotskian Autotelic Agents presented in chapter **??** and **??** was conducted in close collaboration with Cédric Colas (INRIA) who acted as a mentor at the beginning of my thesis, providing me all the tools to carry out efficient research. More specifically, the IMAGINE approach was developed in collaboration with Nicolas Lair (INSERM, Cloud Temple), Peter-Ford Dominey (INSERM), and Jean-Michel Dussoux (Cloud Temple). Finally, our work on grounding spatio-temporal language with transformers (chapter **??**) is the result of a project with Laetitia Teodorescu (INRIA) and her supervisor Katja Hofman (Microsoft Research).

1.2.2 Publications

Journals

- Autotelic Agents with Intrinsically Motivated Goal-Conditioned Reinforcement Learning: A Short Survey, *Journal of Artificial Intelligence Research* 74 (2022), 1159-1199. Colas et al. (2022b) (Co-author)
- Language and Culture Internalisation for Human-Like Autotelic AI, *Nature Machine Intelligence* (2022) Colas et al. (2022a) (Co-first-author)

Conferences

- Language as a Cognitive Tool to Imagine Goals in Curiosity-Driven Exploration, *Advances in Neural Information Processing Systems* 33 (2020). Colas et al. (2020a) (Co-first-author)
- Grounding Spatio-Temporal Language with Transformers, *Advances in Neural Information Processing Systems* 34 (2021). Karch et al. (2021) (Co-first-author)
- Learning to Guide and to Be Guided in the Architect-Builder Problem, *International Conference on Learning Representations* (2022). Barde et al. (2022) (Co-first-author)

Workshops

- Deep Sets for Generalization in RL, *ICLR 2020 workshop Beyond tabula rasa in reinforcement learning: agents that remember, adapt, and generalize*. Karch et al. (2020) (Co-first-author)
- Language-Goal Imagination to Foster Creative Exploration in Deep RL, *ICML 2020 workshop Language in Reinforcement Learning*.

Pre-print

- Emergence of Shared Sensory-motor Graphical Language from Visual Input (2022). Lemesle et al. (2022) (Co-first-author)

Chapter 2

Background: Standard AI Paradigms

Contents

Our contributions bridge standard AI paradigms and developmental psychology to investigate two fundamental research questions (1) the language acquisition problem (self-organisation of cultural conventions) and (2) the open-ended skill acquisition problem (self-organisation of trajectories). In this chapter, we will first present standard AI problems and their associated families of algorithmic solutions before getting into the specifications of the two problems we investigate.

2.1 Reinforcement Learning

Problem

In a Reinforcement Learning problem, an agent learns to perform sequences of actions in an environment by maximizing some notion of cumulative reward (Sutton & Barto, 2018). The agent interacts with the environment in the form of a temporal sequence unfolding from time $t = 0$ to time $t = T$, T being the episode horizon and representing the lifetime of the agent (potentially variable or infinite). RL problems are commonly framed as Markov Decision Processes (MDPs).

Definition

Markov Decision Process (MDP):

$$\mathcal{M} = \{\mathcal{S}, \mathcal{A}, \mathcal{T}, \rho_0, R\} \tag{2.1}$$

where $\mathcal{S}$ and $\mathcal{A}$ are respectively the state and action spaces, $\mathcal{T}$ is the transition function that dictates how actions impact the world (lead to the next state), ρ_0 is the initial state distribution and R is the reward function.

At the beginning of an episode, the agent starts in the initial state $s_0 \sim \rho_0(\mathcal{S})$. At each time step the agents takes action $a_t \in \mathcal{A}$ and observes the next state $s' = s_{t+1} \in \mathcal{S}$ and the reward $r_{t+1} = R(s_t, a_t)$. A diagram of interaction is given in Fig. 2.1. The transition

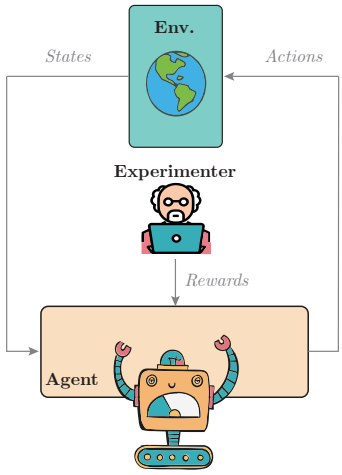


Figure 2.1: Interactions in a RL loop

function $\mathcal{T}$ gives the distribution of the following states from the current state and action: $\mathcal{T} = P_E(.|s, a)$ with P_E being the (potentially stochastic) dynamics of the environment. In an MDP, the transition function must respect the *Markov property*: a future state (s') must only depend on the current state (s) and not on its predecessor, i.e. the transition function is memoryless.

$$P_E(s_{t+1}|s_t, a_t) = P_E(s_{t+1}|s_0, \ldots, s_t, a_t) \tag{2.2}$$

In a RL problem, the behavior of the agent is expressed as a *policy* $\pi : \mathcal{S} \to \mathcal{A}$ that predicts the next action a based on the current state s. This policy can be stochastic $a_t \sim \pi(.|s_t)$ or deterministic $a_t = \bar{\pi}(s_t)$. When agents interact in an environment, they produce *trajectories*. A trajectory is a sequence of states and actions $\tau = (s_0, a_0, \ldots, s_T, a_T)$. When both the dynamics of the environment and the policy of the agent is stochastic, the probability of a trajectory is:

$$P(\tau|\pi) = \rho_0(s_0) \prod_{t=0}^{T-1} P_E(s_{t+1}|s_t, a_t)\pi(a_t|s_t) \tag{2.3}$$

The objective of the agent is to maximize the cumulative reward computed over trajectories (R^{tot}). When computing the aggregation of rewards, we often introduce discounting and give smaller weights to delayed rewards. The return of a trajectory is therefore:

$$R^{tot}(\tau) = \sum_{t=0}^{T} \gamma^t R(s_t, a_t) \tag{2.4}$$

with $\gamma \in]0, 1]$ being a constant discount factor. We call the optimal policy π^*, the behavior that maximizes the expected return:

$$\pi^* = \underset{\pi}{\text{argmax}} \underset{\tau\sim\pi}{\mathbb{E}}[R^{tot}(\tau)] = \underset{\pi}{\text{argmax}} \underset{(a_t\sim\pi, s_t\sim P_E)}{\mathbb{E}} \left[\sum_{t=0}^{T} \gamma^t R(s_t, a_t)\right] \tag{2.5}$$

The reward function plays therefore a crucial role in a RL problem as its maximization will directly shape the behavior of the agent.

Value Functions

Most RL algorithms rely on the definition of *value* and *action-value* functions:

Definitions

- The Value Function $V_\pi(s)$ of a policy π gives the expected return of a trajectory starting from s and following π.
- The Action-value Function $Q_\pi(s,a)$ is the expected return of the trajectory taking action a from state s before following π from the next state s'.

Action-value functions are powerful because they allow us to instantly assess the quality of a situation without waiting for the end of the trajectory. The value and action-value function obey the Bellman expectation equations (Sutton et al., 1998), a recursive definition that states that the value of a certain state (when following policy π) is equal to the sum of the instantaneous reward and the value from the next state.

$$\begin{cases} V_\pi(s) = \mathop{\mathbb{E}}_{(a\sim\pi, s'\sim P_E)}[R(s,a) + \gamma V_\pi(s')] \\ Q_\pi(s,a) = \mathop{\mathbb{E}}_{s'\sim P_E}\Big[R(s,a) + \gamma \mathop{\mathbb{E}}_{a'\sim\pi}[Q_\pi(s',a')]\Big] \end{cases} \tag{2.6}$$

The value and action-value functions also follow the Bellman optimality equation where expectations over actions are replaced by max operators.

$$\begin{cases} V^*(s) = \max_a \mathop{\mathbb{E}}_{s'\sim P_E}[R(s,a) + \gamma V^*(s')] \\ Q^*(s,a) = \mathop{\mathbb{E}}_{s'\sim P_E}\Big[R(s,a) + \gamma \max_{a'}[Q^*(s',a')]\Big] \end{cases} \tag{2.7}$$

Acting greedily with respect to the optimal action-value function gives the optimal policy:

$$\pi^*(s) = \operatorname*{argmax}_a Q^*(s,a) \tag{2.8}$$

Computing Q^* is therefore a way to solve a RL problem. When agents have access to perfect knowledge of the dynamic of the environment (P_E) and when the dimensionality of $\mathcal{S}$ and $\mathcal{A}$ is small, they can do planning to find the optimal action-value function via Dynamic Programming (Bellman, 1966) for instance. Planning approaches that leverage the transition function of the environments are called *model-based* RL algorithms. They are opposed to *model-free* RL algorithms that do not use P_E but interact directly with a simulator (with transition function P_S).

Because the present research builds on both families of solutions, we detail the techniques used for each in the following paragraphs. We first briefly detail the *Monte-Carlo Tree Search* planning algorithm (MCTS) (Browne et al., 2012) used in our first experimental contribution (in chapter 5) and then introduce the deep RL algorithm used in chapter **??**.

Model-based rl with mcts:

MCTS is a tree-search algorithm that seeks to identify the optimal policy by finding the action with the highest Q-value. To this end, MCTS build an estimate $\hat{Q}(s,a)$ for $a \in \mathcal{A}$ in

a given state s and acts greedily with respect to this estimate. Each node of the tree is a state s while edges are the potential actions. The MCTS algorithm grows the tree iteratively using an exploration/exploitation tradeoff to efficiently refine $\hat{Q}$ in promising regions of the MDP. More specifically, each iteration of the MCTS algorithm contains four steps:

1. **Selection:** In the selection phase, the MCTS algorithm starts from the root node and uses a tree policy to decide which node to expand. The tree policy is guided by an evaluation function (UCT) and stops when a node with remaining actions to explore is reached.
2. **Expansion:** Once a leaf node is reached, a new action a is sampled among the non-explored ones and the corresponding node is computed using the transition function $s' \sim P_E(.|s,a)$
3. **Simulation:** From the newly created node corresponding to state s', a simulation policy π_{sim} is used to draw a full trajectory (until termination or for a predefined horizon) and compute return R^{tot}. π_{sim} is often a random policy.
4. **Backpropagation:** R^{tot} is backpropagated to the root node as indicated in Fig. 2.2.

For the tree policy evaluation function, we use the Upper Confidence Bound (Auer et al., 2002): $UCT = \frac{1}{k}\sum_{i=0}^{k} R_i^{tot} + C\sqrt{\frac{\ln(n)}{k}}$ where k is the number of completed trajectory going through node s and n is the number of iterations. The first term of UCT is an estimation of the expected return while the second term encourages the tree policy to explore unexpanded nodes.

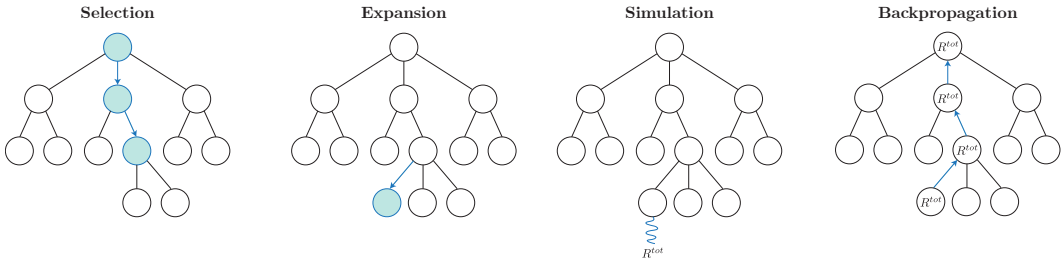


Figure 2.2: The four steps of an MCTS iteration

Model-free rl with Q-learning:

Some of the experimental contributions of this research build on the *Deep Deterministic Policy Gradient* (DDPG) algorithm (Lillicrap et al., 2016). DDPG derives from *Deep Q-Networks* (DQN) (Mnih et al., 2015) which is itself a deep learning implementation of the standard *Q-learning* algorithm (Watkins & Dayan, 1992). In this paragraph, we propose to detail the steps that allow building DDPG from Q-learning.

Q-learning is an *off-policy* RL algorithm. Off-policy algorithms, in contrast to on-policy algorithms, learn to approximate the action-value Q^* of an optimal policy independently of the policy used for data collection. Q-learning relies on transitions (s,a,r,s') collected by a policy π_c interacting with a simulator P_S. Assuming that Q is a linear combination of

features (ϕ): $Q(s,a;\theta) = \theta^T\phi(s,a)$, the algorithm iteratively learns to approximate Q^* by minimizing the temporal difference error (TD-error):

$$\mathcal{L}_i = \mathop{\mathbb{E}}_{(s\sim P_S, a\sim\pi_c)}\left[(y_i - Q(s,a;\theta_i))^2\right] \text{ with } y_i = \mathop{\mathbb{E}}_{s'\sim P_S}\left[r + \gamma \max_{a'} Q(s',a';\theta_{i-1})\right] \tag{2.9}$$

In the original formulation of the Q-learning algorithm by Watkins & Dayan (1992), they consider a tabular setting and store the Q-values at each iteration in a table ($Q_i[s,a]$) instead of using linear function approximations. The update of the table writes:

$$Q_{i+1}[s,a] \leftarrow Q_i[s,a] + \alpha\left(r + \gamma \max_{a'} Q_i[s',a'] - Q_i[s,a]\right) \tag{2.10}$$

dqn proposes to represent the action-value function with deep neural networks: $Q(s,a;\theta)$ with parameters θ. The architecture of the network takes a state s as input and outputs the value of each action $Q(s,a)\forall a \in \mathcal{A}$. Thus DQN only works with discrete action space. When differentiating Eq. (2.9) with respect to the neural network parameters, we get:

$$\nabla_{\theta_i}\mathcal{L}_i(\theta_i) = \mathop{\mathbb{E}}_{(s\sim P_S, a\sim\pi_c)}\left[(y_i - Q(s,a;\theta_i))\nabla_{\theta_i}Q(s,a;\theta_i)\right] \tag{2.11}$$

During differentiation, one has to pay particular attention to freezing the weights of the network when evaluating y_i. Deep neural networks are known to exhibit training instabilities. In order to stabilize learning, Mnih et al. (2015) proposed two main innovations:

- *Experience Replay*: The agent uses a replay buffer to store transitions during interactions. During learning, the transitions are then sampled uniformly to perform updates. This enables breaking the correlation between successive transitions and reusing them.
- *Target network*: A target network is used to compute target y. This network is initialized with the actual Q-network ($Q_{targ}(s,a;\theta_{targ}) = Q(s,a;\theta)$ but updated less frequently than the actual Q-network. Updates are often performed using *Polyak averaging* (Polyak & Juditsky, 1992): $\theta_{targ} \leftarrow \rho\theta_{targ} + (1-\rho)\theta$ with ρ being the polyak factor.

ddpg is an adaptation of DQN to continuous action space. The challenge of dealing with continuous actions is to act greedily with respect to the learned Q-value. i.e. to evaluate $\text{argmax}_a\, Q(s,a)$. To overcome this, DDPG concurrently learns a deterministic policy with the Q-function. This policy is a parametrized network $\pi(s;\phi)$ with parameters ϕ and is obtained by gradient ascent. Moreover, since $\pi(s;\phi) \approx \text{argmax}_a\, Q(s,a,\theta)$ it can be injected in Eq. (2.9). We, therefore, have the two following losses to optimize:

$$\begin{cases} \mathcal{L}_{\pi_\phi} = \mathop{\mathbb{E}}_{(s\sim P_S)}\left[Q_\theta(s,\pi_\phi(s))\right] & \text{(Policy loss)} \\ \mathcal{L}_{Q_\theta} = \mathop{\mathbb{E}}_{(s\sim P_S, a\sim\pi_c)}\left[(y - Q_\theta(s,a))^2\right] & \text{(Q-value loss)} \\ \quad \text{with } y = \mathop{\mathbb{E}}_{s'\sim P_S}\left[r + \gamma Q_\theta(s',\pi_\phi)\right] \end{cases} \tag{2.12}$$

where parameter dependencies have been subscripted.

Other model-free rl algorithms

There are numerous algorithms within the field of DRL, including on-policy methos like TRPO (Schulman et al., 2015), PPO (Schulman et al., 2017) as well as more advanced off-policy approaches like TD3 (Fujimoto et al., 2018) and SAC (Haarnoja et al., 2018).

2.2 Imitation Learning

Problem

Imitation Learning (IL) (Pomerleau, 1988; Schaal, 1996; Osa et al., 2018) is a field that considers an agent learning in a MDP in which the reward function is not explicitly defined, but where the agent can observe demonstrations of the task it is intended to perform. IL is particularly useful in situations where it is difficult for the experimenter to design a task-specific reward function, but demonstrations are available. A classic example from the literature is the application of IL to self-driving cars. It is impractical to specify a reward function for the task of driving as successful drivers constantly adjust their criteria to adapt to the various events that occur on the road. However, there is a vast amount of video footage of people driving that could potentially be utilized by the agent to learn.

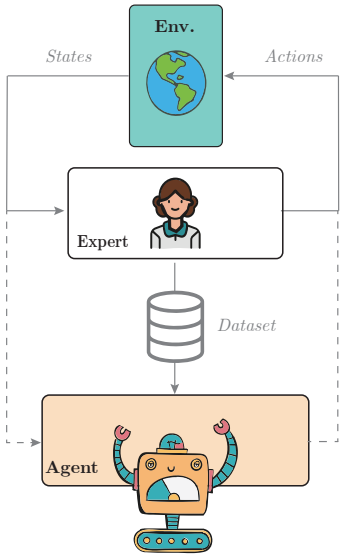


Figure 2.3: Interactions in a IL problem. The agent never interacts with the environment during learning but can interact with it to test its behavior (dashed lines).

A standard way of formalizing the IL problem is to find a policy that minimizes the divergence between the expert and learner data distribution. Provided a dataset $\mathcal{D} = \{(\tau_i)\}_{i=1}^{N}$ containing expert trajectories of features $\tau = [\phi_0, \ldots, \phi_T]$. If $q_{\pi^*}(\phi)$ is the distribution of features induced by the expert's policy (supposed optimal π^*) and $p_\pi(\phi)$ is the distribution of features induced by the learners' policy (π), the goal of IL is to find policy $\hat{\pi}$ such that:

$$\hat{\pi} = \underset{\pi}{\operatorname{argmin}}\, D(q_{\pi^*}(\phi), p_\pi(\phi)) \tag{2.13}$$

with D being a measure of differences between probability distributions such as the well-known Kullback-Leibler (KL) divergence.

Behavioral Cloning

An intuitive way of solving an IL problem is to frame it as a supervised learning setting and do *Behavioral Cloning*(BC). Given a dataset of trajectories $\mathcal{D} = \{(\tau_i)\}_{i=1}^{N}$ with $\tau = [(s_0, a_0) \dots (s_T, a_T)]$, one directly minimizes the cross entropy loss:

$$\mathcal{L}_\pi = - \mathop{\mathbb{E}}_{(s,a) \sim \mathcal{D}} [\log \pi(s, a)] \tag{2.14}$$

Minimizing this cross-entropy loss is in fact equivalent to minimizing the KL-divergence between the trajectory distribution of the expert $P(\tau|\pi^*)$ and the trajectory distribution of the learner $P(\tau|\pi)$ (Ke et al., 2020):

$$D_{KL}(P(\tau|\pi^*), P(\tau|\pi)) = \sum_{\tau \in \mathcal{D}} P(\tau|\pi^*) \log \left(\frac{P(\tau|\pi^*)}{P(\tau|\pi)} \right) \tag{2.15}$$

Injecting the definition of the trajectory distribution of Eq. (2.3) we get that:

$$D_{KL}(P(\tau|\pi^*), P(\tau|\pi)) = \sum_{\tau \in \mathcal{D}} P(\tau|\pi^*) \log \left(\prod_{t=0}^{T-1} \frac{\pi^*(a_t|s_t)}{\pi(a_t|s_t)} \right) \tag{2.16}$$

$$= \sum_{\tau \in \mathcal{D}} P(\tau|\pi^*) \sum_{t=0}^{T-1} (\log \pi^*(a_t|s_t) - \log \pi(a_t|s_t)) \tag{2.17}$$

$$= \mathop{\mathbb{E}}_{(s,a) \sim \mathcal{D}} [\log \pi^*(a_t|s_t) - \log \pi(a_t|s_t)] \tag{2.18}$$

We will use behavioral cloning in chapter 5. BC is a straightforward method for reproducing expert behavior. However, simple BC only works if the agent operates in the same region of the state space as the states provided in $\mathcal{D}$. Otherwise, the policy of the learner will progressively deviate from this region accumulating errors at each time step. This compounding error is called *distributional mismatch*. One way of addressing it is to iteratively collect new expert data when needed (in the initially uncovered region of the state space) (Ross et al., 2011).

Another limitation of BC is that it is only able to derive an optimal policy from optimal expert trajectories, meaning that the learned policy will not exceed the performance of the expert. In some applications collecting optimal trajectories is not always possible. As a result, some researchers have turned to *Inverse Reinforcement Learning* (IRL) as an alternative approach.

Inverse Reinforcement Learning

Similar to RL, IRL can be understood both as a problem and a category of techniques. The IRL problem consists in recovering the reward function of an expert given a dataset of its trajectories (Ng & Russell, 2000). As such IRL algorithmic solutions followed by RL can form a solution to the IL problem. The combination of IRL followed by RL is called *Apprenticeship Learning* (Abbeel & Ng, 2004). As opposed to BC, apprenticeship learning ensures that the learned policy is bellman consistent (with respect to an underlying learned value-function). As formalized by Klein et al. (2011), there are mainly three categories of strategies to obtain the policy:

1. Feature-expectation-based methods as proposed by Ziebart et al. (2008) which learn a reward function such that the feature expectation of the optimal policy (according to the learned reward function) is similar to the feature expectation of the expert policy.
2. Margin-maximization-based methods (Ratliff et al., 2006), which formulate IRL as a constrained optimization problem in which the expert's examples have a higher expected cumulative reward than all other policies by a certain margin.
3. Approaches based on the parameterization of the policy by the reward (Neu & Szepesvári, 2007): If it is assumed that the expert follows a Gibbs policy (or the optimal value function related to the optimized reward function), it is possible to estimate the likelihood of a set of state-action pairs provided by the expert.

Recent feature-expectation-based approaches use technics similar to generative adversarial networks (GAN) (Goodfellow et al., 2014) to imitate complex behavior in high-dimensional environments (Ho & Ermon, 2016). Other approaches use ranking of trajectories to reach better-than-demonstrator performances (Brown et al., 2020a). As we do not leverage IRL in our contributions we will not detail these methods (see Arora & Doshi (2021) for a thorough survey of IRL algorithms).

2.3 Muli-Goal Reinforcement Learning

Standard RL can be extended to a multi-goal setting. Let us return to the definition of goal by Elliot & Fryer (2008) provided in the introduction (Sec. 1.1.1):

> "*A goal is a cognitive representation of a future object that the organism is committed to approach or avoid* ".

RL algorithms seem, indeed, to be a good fit to train goal-conditioned agents: they 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*').

Formal Definition of Goals and Skills

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:

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 objective of a goal-conditioned agent is to learn a *goal-conditioned policy*: a function that generates the next action given the current state and the goal $a_t \sim \pi(\cdot|s_t, z_g)$. 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.

Definition

- Goal-conditioned policy: a function that generates the next action given the current state and the goal.
- Skill: the association of a goal and a policy to reach it.

Problem

By replacing the unique reward function R by the space of reward functions $\mathcal{R}_{\mathcal{G}}$ in the definition of MDP of Eq. (2.1), 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. $\mathcal{T}, R, \rho_0$, 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. As one can observe in Fig. 2.4, the experimenter sets goals to the agent, and provides the associated reward functions.

Solutions: Horde, UVFA, and HER

Goal-conditioned agents see their behavior affected by the goal they pursue. This is formalized via goal-conditioned policies, that is policies that produce actions based on the environment state and the agent's current goal:

$$\Pi : \mathcal{S} \times \mathcal{Z}_{\mathcal{G}} \to \mathcal{A} \tag{2.19}$$

where $\mathcal{Z}_{\mathcal{G}}$ is the space of goal embeddings corresponding to the goal space $\mathcal{G}$ (Schaul et al., 2015). Note that ensembles of policies can also be formalized this way, via a meta-policy

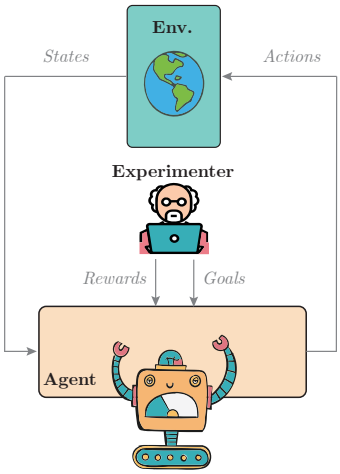


Figure 2.4: Interactions in a multi-goal RL loop. The experimenter provides goals and their associated rewards to the agent.

Π that retrieves the particular policy from a one-hot goal embedding z_g (Kaelbling, 1993; Sutton et al., 2011).

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. (2015) 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., 2017). 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.

A typology of Goal Representations

The goal concept and multi-goal RL will be a central aspect of the autotelic RL framework that we detail in Sec. 3.3. Therefore we, here, propose to review the different kinds of goal representations found in the literature. For each category of goal, we detail the form of the goal embedding and the reward function.

Goals as choices between multiple objectives. Goals can be expressed as a list of different objectives the agent can choose from. This is the case in Oh et al. (2017);

Mankowitz et al. (2018); Codevilla et al. (2018); Chan et al. (2019b).

Goal Embedding	*Reward Function*
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: $z_g^i = (\mathbb{1}_{j=i})_{j=[1..N]}$.	The goal-conditioned reward function is a collection of N distinct reward functions $R_{\mathcal{G}}(\cdot) = R_i(\cdot)$ if $z_g = z_g^i$.

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

Goal Embedding	*Reward Function*
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.	$R_{\mathcal{G}}$ is based on a distance metric D. The reward can be dense: $R_g = R_{\mathcal{G}}(s\|z_g) = -\alpha \times D(\varphi(s),\ z_g)$, or sparse: $R_{\mathcal{G}}(s\|z_g) = 1$ if $D(\varphi(s),\ z_g) < \epsilon$, 0 otherwise.

In manipulation tasks, z_g can be target block coordinates (Andrychowicz et al., 2017; Nair et al., 2018a; Plappert et al., 2018; Colas et al., 2019; 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 (Schaul et al., 2015; Florensa et al., 2018). 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; Kova et al., 2020).

Goals as abstract binary problems. Some goals cannot be expressed as target state features but can be represented by *binary problems*, where each goal is expressed as a set of constraints on the state that are either verified or not.

Goal Embedding	*Reward Function*
z_g can be any expression of the set of constraints that the state should respect. Akakzia et al. (2021); Ecoffet et al. (2021) propose a pre-defined discrete state representation. Another way to express sets of constraints is via language-based predicates	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)

When goals are expressed in language, 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., 2019a; Jiang et al., 2019; Bahdanau et al., 2019a,c; Hill et al., 2020; Cideron

et al., 2020; 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*" (see chapter **??**), *relational goals* like "*sort objects by size*" (Jiang et al., 2019), "*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., 2019). 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

Goals as a multi-objective balance. Finally, 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 Embedding	*Reward Function*
z_g are 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.	The reward is expressed as a convex combination of 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.

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 control of the discount factor, effectively controlling the rate at which rewards are discounted as time goes by (Badia et al., 2020a).

2.4 Multi-Agent Reinforcement Learning

Problem

Standard RL can also be extended to scenarios where several agents interact with the environment. For this purpose MDPs are extended to *Markov Games*.

Definition

Markov Game are defined by the following terms:

$$\mathcal{M} = \{\mathcal{S}, \mathcal{T}, \rho_0, \{\mathcal{O}_i, \mathcal{A}_i, R_i\}_{i=1}^{N}\} \tag{2.20}$$

The first three terms of a Markov Game are the same as those of a MDP: $\mathcal{S}$ is the state space, $\mathcal{T}$ is the transition function, and ρ_0 the initial state distribution. However, each agent (denoted by the index i) perceives a different perspective of the state through observation transformation $\mathcal{O}_i$. Agents also have different action spaces $\mathcal{A}_i$ and reward function R_i

In Multi-Agent Reinforcement learning (MARL), each agent aims at learning a policy that maps their observation $o_i = \mathcal{O}_i(s)$ to actions: $a_i \sim \pi_i(\cdot|o_i)$. Similarly to RL, each

agent aim at maximizing its expected return:

$$\pi_i^* = \underset{\pi_i}{\text{argmax}} \underset{(a_t \sim \pi_i, s_t \sim P_E}{\mathbb{E}} \left[\sum_{t=0}^{T} \gamma^t R_i(\mathcal{O}_i(s_t), a_t) \right] \tag{2.21}$$

A diagram of interaction is provided in Fig. 2.5. The field of MARL considers mainly two types of tasks:

- *Cooperative tasks* where the agents pursue the same goal and need to coordinate in order to solve it. Cooperative tasks are usually hard to design and often involve the maximization of a common objective (sometimes at the expense of individual gains). For a review of cooperative MARL see OroojlooyJadid & Hajinezhad (2019).
- *Competitive tasks* where the agents pursue non-aligned goals. In these settings agents explicitly aim at maximizing their individual gains.

Among the recent innovations in MARL, Baker et al. (2020) trained agents to play the hide-and-seek game, Pérolat et al. (2017) to solve common-pool resource problems, and more recently Stooke et al. (2021) trained an agent on a spectrum of cooperative and competitive tasks including cooperative games to find objects, hide and seek or even capture the flag.

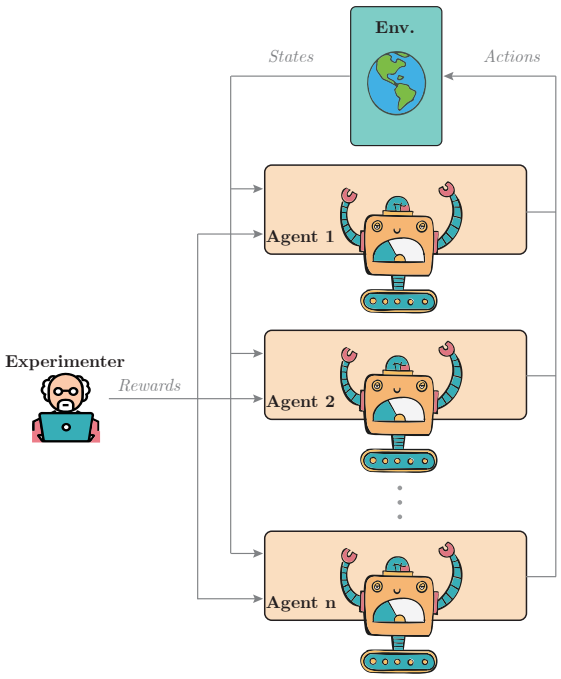


Figure 2.5: Diagram of interactions in a MARL loop. Each agent perceives a (potentially) different perspective of the states provided by the environment. Each agent also has its own action space and is given a (potentially) different reward.

Solution

One of the main challenges of multi-agent learning systems is to take into account the non-stationary dynamics caused by the change of state of the agents when they learn. Indeed,

an isolated agent of a Markov game does not evolve in a stationary MDP because all agents are learning, and their behavior will be different during training. For this reason, most of the MARL algorithms rely on the *centralized training, decentralized execution* paradigm. For instance, Muli-Agent Deep Deterministic Policy gradient (Lowe et al., 2017), uses a centralized training procedure where all agents can see other agents' observations and actions to learn an action-value function that is then used to optimize decentralized policies that only depends on local observations. As none of our contributions builds on MARL we will not elaborate on other MARL algorithms.

Chapter 3

Problem Definition: Developmental AI

Contents

The present research expands upon the standard AI methods presented in the previous chapter, with the aim of investigating fundamental inquiries within the domain of Developmental Artificial Intelligence. Developmental AI is a multidisciplinary field that integrates principles from artificial intelligence, developmental psychology, linguistics, and neuroscience to simulate and analyze the cognitive development of sensorimotor, cognitive, and cultural structures, both at the level of artificial agents and at the level of populations. While standard AI paradigms are structured around precise and formal problems addressed by algorithmic contributions, developmental AI strives to create machine systems that can learn in an autonomous, autotelic, and open-ended manner, similar to the way children learn. In this research the specific questions that we investigate are: 1) How do cultural conventions emerge through interactions among agents in social contexts? 2) How can autotelic artificial agents utilize cultural conventions to acquire open-ended skill repertoires? These questions can be approached through the lens of self-organization theory. Specifically, this study will examine: 1) the self-organization of conventions, or language, among agents, and 2) the role of cultural conventions in the self-organization of agents' developmental trajectories.

The initial part of this section (Sec. 3.1) outlines the fundamental concept of self-organization, encompassing the development of cultural conventions and the formation of individual trajectories. A typology of the language formation problem is then presented in Sec. 3.2, which provides us a structured approach to categorizing our two first research investigations (presented in Sec. 3.2.2), namely the self-organization of graphical sensorymotor language, and the formation of cultural convention in the Architect-Builder problem.

These contributions will be developed in Part I of this manuscript. Then, Sec. 3.3 describes the open-ended formation of skill repertoire and introduces the autotelic RL framework. The autotelic RL framework will serve as a basis to explore the role of cultural conventions in the self-organization of developmental trajectories. We outline our contributions in Sec. 3.3.2 and detail them in Part II of this manuscript.

3.1 Self-organization Theory

Self-organization is a term now used in a variety of sciences that can be described with the following definition:

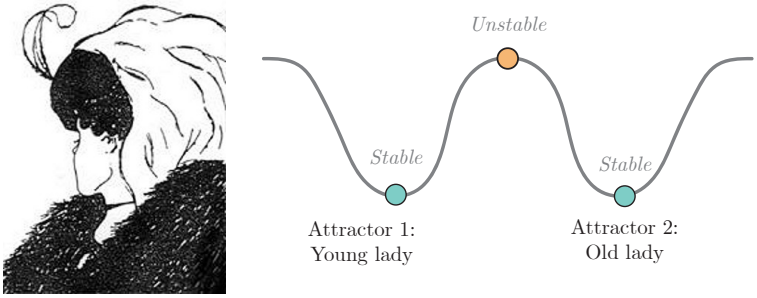
Definition

Self-organization is a process by which spontaneously ordered patterns and structures emerge from the interactions of the many constituents of a system without the need for central control or external guidance. Crucially, the emergent global structure of self-organizing systems has different properties than its local constituents.

Paradoxically, the notion of *emerging order* draws its origin from the study of chaos and was originally used to describe thermodynamical systems that spontaneously organize themself from complex chaotic interactions. The theory of self-organization was formalized by cybernetician Ashby (1962). Borrowing concepts from dynamical system theory, he stated that any complex dynamical systems organize themselves around specific 'attractors' within a vast landscape of possible states. These attractors are stable equilibrium points and may be multiple for a given system. An intuitive explanation of attractors, proposed by Dilts (1995), is given in Fig. 3.1. It illustrates how our complex perception system can fall into different attractors when presented with an illusion. In one case, we see a young woman wearing a necklace looking up to the left, in the other we perceive an old woman leaning slightly forward. This illustration also demonstrates that certain attractors may be difficult to reach. Indeed, when looking at Fig. 3.1 we often rapidly converge to one attractor and have difficulty escaping it to organize our perception around the other. Finding an attractor requires exploring the landscape of possible states. For this, noise and stochasticity can help as described by von Foerster (2003):

> "*I think it is favorable to have some noise in the system. If a system is going to freeze into a particular state, it is inadaptable and this final state may be altogether wrong. It will be incapable of adjusting itself to something that is a more appropriate situation.*"

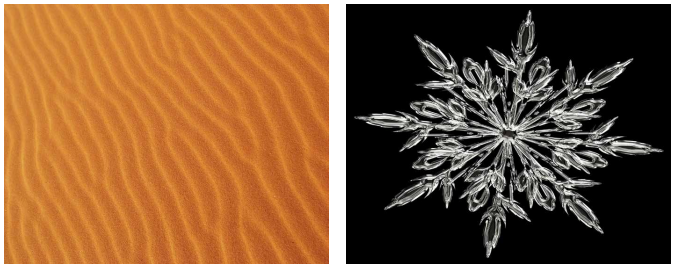
As displayed in Fig. 3.2, nature is full of examples of self-organizing systems. Physical systems exhibit self-organizing behavior through the formation of patterns, such as the longitudinal stripes of sand dunes or the crystalline structure of snowflakes. Similarly, biological systems display self-organizing behaviors(Camazine et al., 2001), as exhibited in the social organization of fish schools and insect swarms, as well as in their ability to collectively adapt to and modify their environment when termites construct mounds and bees build their hives.



(a) (b)

Figure 3.1: (a) My Wife and My Mother-In-Law, by the cartoonist W. E. Hill, 1915 (b) Stability plots illustrating the two attractors of the cartoon as proposed by Dilts (1995)

(a) Physical self-organizing systems



(b)Biological self-organizing systems

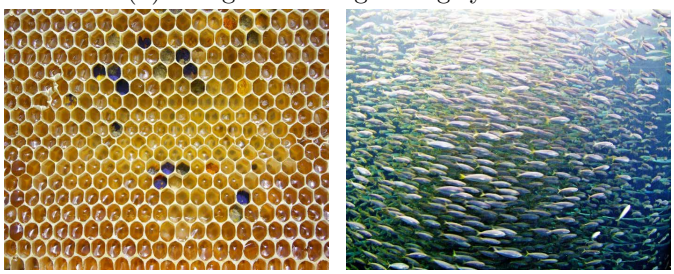


Figure 3.2: Example of self-organizing systems (a) sand dunes in Namibia and crystal structure of snow ice; and (b) a bee comp and a fish school. Images are royalty-free and obtained from pixabay.com

Self-organization also enables creative technical innovation such as the development of self-organizing traffic lights (Ferreira et al., 2010): lights that can adapt to changing traffic conditions through local interactions, rather than relying on communication or external signals. Self-organization is particularly well suited for the problem of traffic light regulation because traffic conditions change constantly. Thus, the problem at hand requires adaptation, a property well captured by self-organization theory.

Self-organization in Developmental AI

Developmental AI problems can be formulated as adaptive problems where one or more agents and the environment are coupled dynamical systems whose interactions are responsible for the agents' behavior (Beer, 1995). In this research, we propose to use the language

of dynamical systems and the theory of self-organization to formalize the two fundamental problems of this research.

First, the problem of the emergence of cultural convention among artificial agents can be analyzed as the self-organization of a language community (Steels, 1995b; Oudeyer, 2005). A cultural convention is thus an attractor of a language community: when multiple agents interact, variations of language behaviors are attracted to an equilibrium state because the more members of a community adopt a particular convention, the stronger the convention becomes. We will present several approaches that model Language formation in Sec. 3.2.

Second, the problem of autonomous skill acquisition can be framed as the self-organization of agents' trajectories where agents use internal mechanisms to develop rather than being controlled by hierarchical top-down control (Pfeifer et al., 2007). In this context, agents develop and grow repertoires of skills via internal drivers and physical interactions with their environment. These internal drivers, referred to as intrinsic motivations (Oudeyer, 2005), allow agents to self-organize their behavior into developmental trajectories and enable them to acquire increasingly complex skills. We will explore the open-ended formation of skill repertoires and present the autotelic approach as a solution to it in Sec. 3.3. We will build on autotelic RL to propose a new vision in developmental RL where agents also leverage social interactions to augment their autonomous learning capabilities

Our two research investigations are at the intersection between the standard AI paradigms (presented in the previous chapter) and fundamental questions within the field of developmental AI. We propose to study them in the next two following questions. Before clearly formalizing our two problems, we survey related implementations at the convergence of standard and developmental AI, namely the computation language formation framework (Sec. 3.2.1) and the autotelic RL problem (Sec. 3.3.1).

3.2 Self-organisation of Cultural Convention: the Language Formation Problem

3.2.1 Computational Models of Language Formation

The study of the origin of language has been a subject of interest and debate among various academic disciplines, including linguistics, archaeology, biology, and anthropology. In this section, we will shortly present the predominant theories on Language formation and explore how artificial agents can help experiment with them. For a thorough review of the synthetic modeling of language origins see Steels (1997). There are three predominant theories on the origin of language:

1. The *Genetic evolution theory* postulates that language, just like biological complexity, is the result of natural selection. According to this theory, humans have an innate language organ inside their brains that contains universal rules helping them learn a language during their development. This claim is backed by the famous poverty of stimulus argument which asserts that children do not observe sufficient data to explain their ability to acquire natural language (Chomsky, 1975). The genetic evolution theory thus implies that there exist language genes that code for the language organ and that language is preserved due to genetic transmission.
2. The *Adaptation and self-organization theory* on the other hand supposes that language is preserved in the memories of individuals and transmitted through cultural and social interactions during imitation and acquisition processes. In the adaptation hypothesis, there is no language organ but rather a variety of cognitive and motor primitives that facilitate language formation.
3. The *Genetic assimilation theory* assumes that language is the result of dual dynamics that both involve cultural and genetic interactions. The genetic assimilation hypothesis is also known as the Baldwin effect. It states that learned behaviors that confer a selective advantage can become genetically encoded over time. The genetic assimilation theory proposes that initially, humans did not have an innate language structure and that the first forms of language were acquired through adaptation only. But, if the speed of language acquisition played a role in selection, genetic assimilation would have facilitated the development of language acquisition devices.

Language formation with Artificial Agents

The study of language emergence can benefit greatly from the utilization of agent-based modeling and simulation (Hurford, 1989; Brighton, 2002; Cangelosi & Parisi, 2002; Steels, 2015; Kirby et al., 2014). *Computational Experimental Semiotics* (Galantucci & Garrod, 2011) is a field that analyzes the numerous factors that contribute to language emergence by examining a population of simulated agents engaging in two distinct types of interaction: *linguistic* and *genetic interactions*. When two agents take part in linguistic interaction, they are in turn speakers and listeners and respectively produce and receive messages describing a context. To study the formation of meanings, linguistic interactions occur within physical environments that contain objects and embodied situations (Steels & Loetzsch, 2012).

Depending on the communicative success of linguistic interactions, agents can update their internal state and adapt to their artificial peers. To investigate the impact of population dynamics, the studied population is open: new agents enter, and others leave. These new agents, generated through genetic interactions and subject to potential mutations, introduce an element of novelty into the system. Finally, in order to obtain realistic models, the population should be studied as a distributed multi-agent system, i.e. there should not be any main global agent that acts over the entire population. Moreover, just like humans cannot enter the brain of others, agents should not be able to access each other's internal states. A diagram of interactions as well as a high-level algorithmic implementation of the Language formation framework is provided in Fig. 3.3 and Alg. 1.

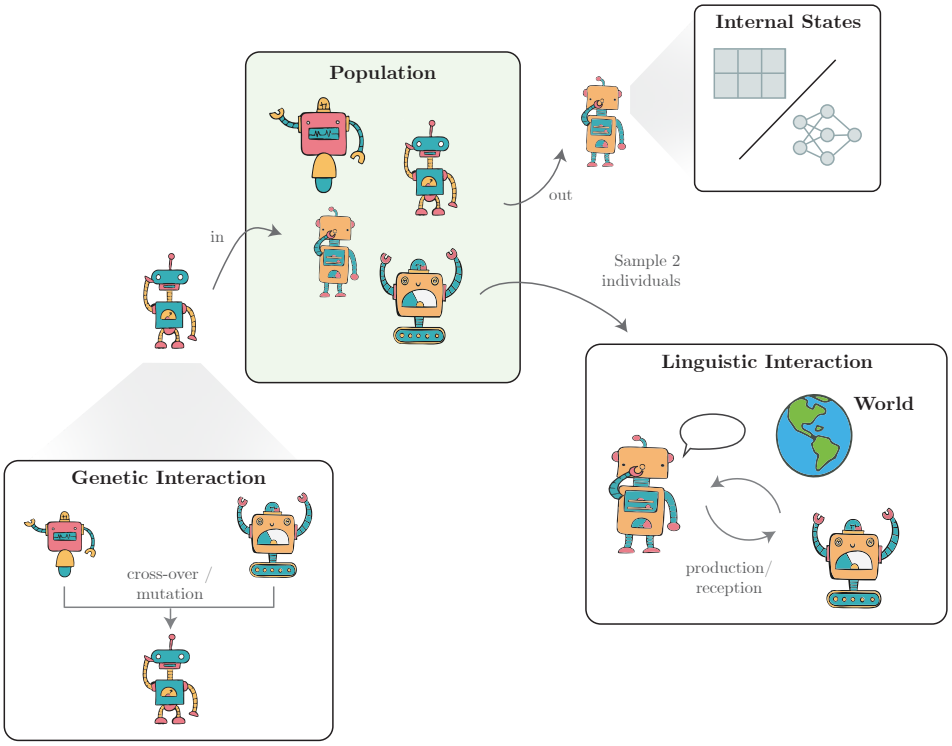


Figure 3.3: The Language formation framework: A population of agents is an open multi-agent system where new agent enters and others leave. New agents are generated by genetic interactions: crossovers between parents with potential mutations. Agents have internal states allowing them to map signals to actions. They can perform linguistic interactions, i.e. exchanging messages to describe a physical situation of the world.

This study will not investigate the impact of population dynamics on the emergence of language. Rather, it will focus on the development of agents during a lifetime and disregard any genetic interactions. We will thus focus on the self-organization of cultural conventions during linguistic interactions. We will restrict our analysis to the smallest population of two individuals.

Algorithm 1: Language formation Simulation

Require: Language Interaction L, Genetic Interaction G, Environment $\mathcal{E}$
  Initialize Population $\mathcal{P}_A$ and internal states of agents
  loop
    Sample two agents from population: $(A_1, A_2) \sim \mathcal{P}_A$
    Store result of linguistic interaction about the world: $\leftarrow L(A_1, A_2, \mathcal{E})$
    Update A_1 and A_2 based on score s
    With prob p_{out}:
      Remove agent from population: $\mathcal{P}_A$.pop()
    With prob p_{in}:
      Sample two parents from population: $(A_1, A_2) \sim \mathcal{P}_A$
      Perform Genetic Interaction: $A' \leftarrow G(A_1, A_2)$
      Add child to population: $\mathcal{P}_A$.add(A')
  end loop

Language Games

The simplest forms of linguistic interaction are coined language games. They derive from *Signaling Games* introduced by Lewis (1969) as a game theoretic approach to the problem of the emergence of conventions. In game theoretic words, a convention is a system of arbitrary rules that enables two players to share meaningful information. Fig. 3.4 presents a simple example of a Lewis game. The two players of a signaling game are the speaker and the listener. In our example, the world is providing two world states to the speaker (w_1 and w_2). Based on the world state, the speaker sends a signal to the listener. Here, there are two available signals (s_1 and s_2). From the received signal, the listener then chose an action (among two actions a_1 and a_2). If the listener picks the correct action for the associated word state then both agents perceive a reward. Note that the Listener never perceives the world state.

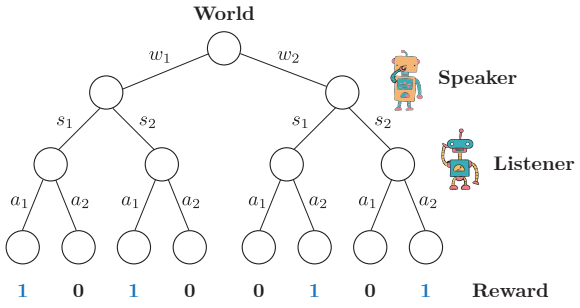


Figure 3.4: Illustration of Lewis Signalling game with two world states, two signs, and two actions.

To investigate the self-organization of conventions around meanings in a more realistic scenario, Steels & Loetzsch (2012) proposed to update signaling games with grounding elements. In a grounded language game, the speaker and the listener are given a shared *context* made of several *referents* (objects) as displayed in Fig. 3.5. The speaker samples a target referent from the context and produces an utterance to name it. Then, the speaker receives the utterance and picks a referent inside the context. If the chosen referent matches

the target referent, the game is a success. To self-organize a language, a population of artificial agents needs to play numerous language games. In doing so, agents will alternate between speakers and listers. Depending on the outcome of the game they will update their internal states to reinforce successful conventions and diminish unsuccessful ones. Note that several update strategies are possible. They vary in how the outcome is actually perceived by the agents. On the speaker side, the referent ground truth (target) is known so the outcome of the game can be directly used for the update. On the other hand, since the listener does not know about the target referent some implementations of language games do not communicate the outcome to the listener. In Steels (2001)'s formulation of language game, the outcome is communicated to the speaker via a retroactive pointing mechanism. The speaker basically points toward the target referent at the end of the game to communicate the outcome to the speaker.

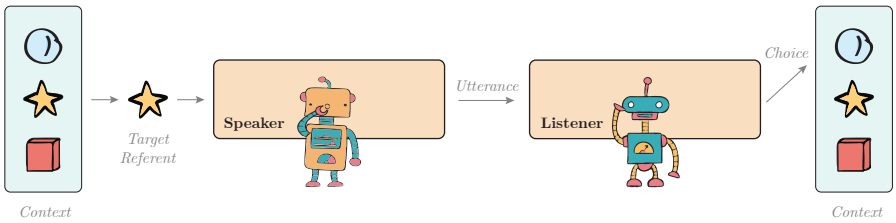


Figure 3.5: Diagram of interactions in a language game

Early solutions to the language game (Steels, 1995b; Oliphant & Batali, 1997; Kirby, 2001) use tables scoring associations between referents and utterances. Given fixed predefined numbers of utterances and referent categories, the agents can adjust the score of utterance/referent association depending on their communicative success. Examples of such tables for the speaker (left) and for the listener (right) are given in Fig. 3.6. If predefined referent categories are not available to the agents, Steels & Loetzsch (2012) propose mechanisms to map visual inputs to object categories. Similarly, the Talking Head experiments (Steels, 2015) propose strategies to adapt the language game to more realistic configurations with flexible and dynamic inventories of words and meanings.

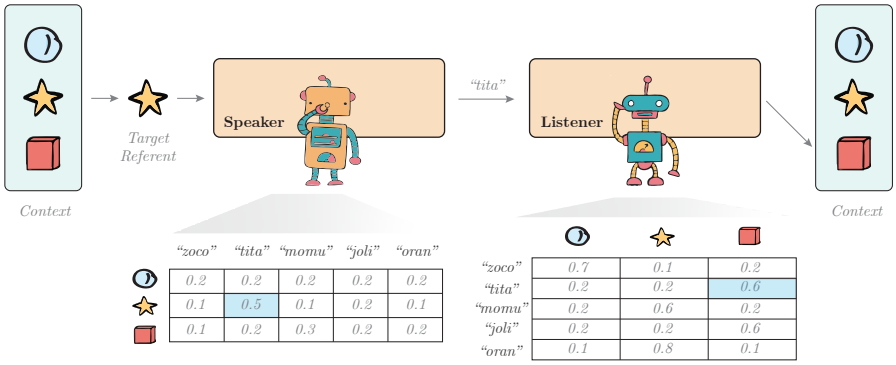


Figure 3.6: Example of agents' tabular internal models, with 3 referents and 5 words.

Neural Communicating Agents

Inspired by the success of Convolutional Neural Network in Computer Vision, Lazaridou et al. (2017) proposed to extend language games to image referents with agents using neural networks to take actions [1]. Fig. 3.7 illustrates their setup. The context is made of two images (i_1 and i_2), a target (t) and a distractor. The utterances are discrete utternaces u coming from a fixed-sized dictionary $\mathcal{V}$. The speaker's utterance is given by a neural network parametrizing a policy that maps the two images to the utterance: $u = \pi_S(i_1, i_2, t; \theta_S)$. Similarly, the listener uses policy π_L to make a choice given the utterance: $a = \pi_L(i_1, i_2, \pi_S(i_1, i_2, t; \theta_S); \theta_L)$. The policies are trained using RL (Sec. 2.1) with reward function R returning 1 iff $\pi_L(i_1, i_2, \pi_S(i_1, i_2, t; \theta_S); \theta_L) := t$. Note that in their implementation the reward and thus the outcome of the game is communicated to both agents which is equivalent to Steel's pointing mechanism.

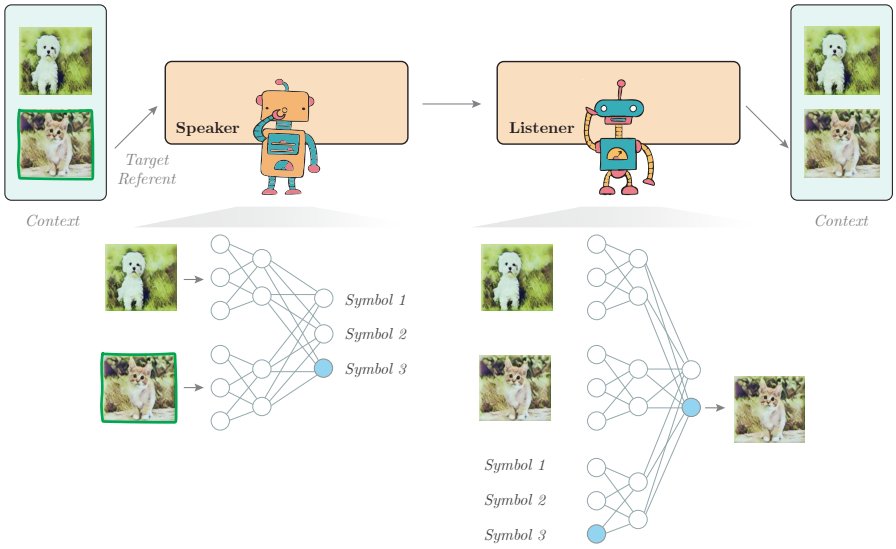


Figure 3.7: Example of agents' neural network internal models, with 2 referents and 3 words (adapted from Lazaridou et al. (2017)).

Beyond scaling previous language game approaches to visual referents, Lazaridou et al. (2017) proposes a strategy to ground the agent's code in natural language. The strategy consists in leveraging a dataset of (image, natural language label) pairs and alternating between RL in the classical language game and standard supervised learning for image classification. In a similar study, Havrylov & Titov (2017) examine the emergence of communication with sequences of symbols and visual referents. They use LSTMs within the speaker and listener architectures to support the encoding and decoding of discrete tokens arranged in fixed-size sequences. They also analyze the compositionality and variability of the emerging sequences in both tabular-rasa and natural language communication.

The identification of the factors that contribute to the emergence of compositional communication code is a fundamental objective within the field of computational linguistics.

[1] In deep learning, language games are often referred to as referential games or guessing games

To this end, the use of neural communicating agents in language games has emerged as a valuable experimental setting. Kottur et al. (2017) propose to analyze how utterances consisting of sequences of symbols can name referents that are compositions of abstract attributes (represented as one-hot vectors). The decomposition of referents into pre-defined hardcoded attributes enables a more comprehensive and systematic analysis of the compositional properties of the evolving communication code. Building upon this work, Chaabouni et al. (2020) emphasize the importance of separating the compositional generalization capabilities of agents and compositional properties of the emerging code. They have established that the former can be achieved independently of the latter. Other works look at the environmental and internal factors that favor the emergence of compositionality. For instance, Rodríguez Luna et al. (2020) show that auxiliary objectives incentivizing object consistency or least effort (the generation of short sequences) support the emergence of compositional code in language games. Similarly, Mu & Goodman (2021) demonstrate that agents solving a variation of the language games where referents are organized in sets of objects agree on a more interpretable and systematic communication code. Finally, Ren et al. (2020) proposed to study the emergence of compositional code in a more complete setting with a population of agents playing language games over several generations.

Goal-Directed Communicating agents

The prior paragraph demonstrates that guessing interactions provide an effective experimental testbed to study Language formation. But, as outlined in the introduction, human language serves a multitude of purposes beyond mere object guessing. Therefore, AI researchers have aimed to examine the development of communication in more realistic scenarios, where agents must communicate to accomplish a collaborative task in complex environments that involve interactions with the physical world across multiple time steps. These problems are modeled using MARL as described in Sec. 2.4. The agents must concurrently learn to interact with the world and communicate with others by observing rewards related to their collaborative goal, provided by an expert. See Fig. 3.8 for a visual representation of these interactions. Seminal works on MARL involving communicating agents consider problems such as efficient car coordination at traffic junctions to avoid collision (Sukhbaatar et al., 2016) or riddles where agents need to combine environmental inputs with information communicated over several time steps to succed (Foerster et al., 2016). In their work, Foerster et al. (2016) introduce two approaches for learning to communicate in MARL: Differentiable Inter-Agent Learning (DIAL) and Reinforced Inter-Agent Learning (RIAL). DIAL is based on the centralized training and decentralized execution method and enables gradient to be exchanged between agents, thereby breaking the assumption of the Language formation framework that agents should not be able to have access to each other's internal states. Conversely, in RIAL, messages are viewed as actions produced by a RL algorithm where each agent treats others as a part of the environment, without the need to have access to other agents internal parameters or to back-propagate gradients. The RIAL algorithm now serves as a baseline for a variety of MARL communication investigations. Jiang & Lu (2018) extended it with an attention mechanism that enables agents to learn when communication is required to solve collaborative tasks. Similarly, Eccles et al. (2019) showed that adding positive signaling (messages must be different in different situations) and positive listening (actions must be different when messages are different) biases to agents via auxiliary

losses yields an increase in communicative performance. For a complete survey of emergent communication in MARL setups, see Zhu et al. (2022).

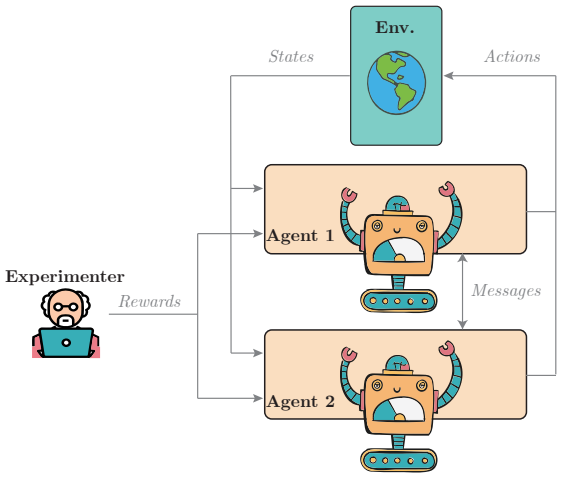


Figure 3.8: Diagram of interactions in MARL emergence communication.

Summary

In this section, we presented the Language formation framework which study the emergence of communication inside a population of agents interacting within linguistic and genetic interactions. Our contributions focus on linguistic interactions and ignore the influence of population dynamics on the emergence of communication. We, therefore, presented the most broadly studied linguistic interactions: the language game. Additionally, we showed that this particular guessing interaction setup could be scaled to neural agents. Finally, we noted that MARL offers a valuable framework to study the emergence of communication as a tool to achieve collaborative behaviors in physically complex environments.

3.2.2 Problem Definition

It this now time to turn to our contributions and to formally pose the specific inquiries we target within the context of artificial communicating agents.

Emergence of Graphical Sensory-motor Communication

Our first contribution **Todo:***add chapter ref* extends the neural communicating agent framework to consider communication in visual language games via a sensory-motor channel. As reviewed in the previous section, prior approaches focused on agents communicating via an idealized communication channel, where utterances (made of a single or a sequence of symbols) are produced by a speaker and 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). Motivated by this observation

we investigate whether artificial agents can develop a shared language in an ecological setting where communication relies on such sensory-motor constraints. To this end, we introduce the *Graphical Referential Game*(GREG) where a speaker must produce a graphical utterance to name a visual referent object while a listener has to select the corresponding object among distractor referents, given the delivered message as illustrated in Fig. 3.9. The utterances are drawing images produced using dynamical motor primitives combined with a sketching library. The referents are images of MNIST (LeCun et al., 1998) digits randomly positioned in the image. The utilization of sensory-motor systems to examine the development of language dates back to the investigation of the origins of digital vocalization systems in the early 2000s de Boer (2000); Oudeyer (2005); Zuidema & De Boer (2009). However such studies were not conducted in grounded language games. They employed imitation games focusing on the observation of the formation of speech utterances, such as syllables and words, through the systematic combination of lower-level meaningless elements (phonemes). In our study, we chose to focus on a drawing system because 1) conversely to models of vocalization, there is a large number of tools available to researchers to implement realistic sketching mechanisms and 2) it has 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).

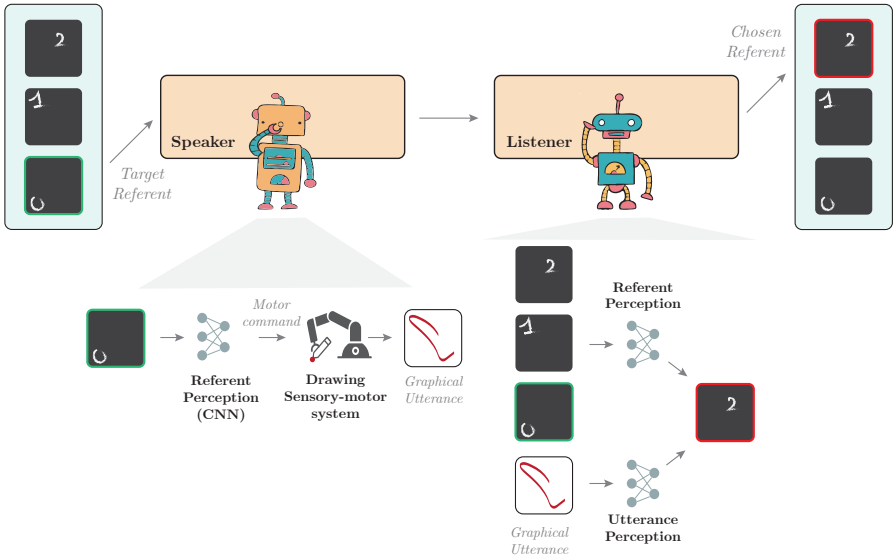


Figure 3.9: The Graphical Referential Game

Studying the GREG we aim at investigating whether a pair of agents can self-organize a shared lexicon from the continuous non-linear constraints of the sensory-motor system. We then propose to study the structure of the emerging signals. We use topographic measures based on a geometric distance to quantify the coherence of the emerging lexicon. Informed by Chaabouni et al. (2020)'s study on the non-equivalence between compositional performance and compositional language, we propose to study these two questions separately. More precisely, we evaluate the communicative generalization performances of our system on referents that are the composition of MNIST digits. We end our study investigating the compositional structure of emerging signs using the same geometry measure as for the coherence.

The Architect-Builder Problem

Our second contribution proposes to study the *Architect-Builder problem* (ABP), a new AI paradigm that studies the goal-directed emergence of communication in a setup where the reward function is not accessible to all agents. The ABP involves two agents, referred to as the *Architect* and the *Builder*, who must collaborate to accomplish a task. Both agents observe the environment state but only the architect knows the goal at hand. The architect possesses knowledge of the goal and is able to receive the reward associated with it, but is unable to take actions in the environment. In contrast, the builder has no knowledge of the goal or reward and is the only agent that can take actions in the environment. In this asymmetrical setup, the architect can only interact with the builder through a communication signal (messages).

The introduction of the ABP aims to address a gap in the existing literature on goal-directed communication with neural agents. Current MARL models typically assume the presence of a centralized rewarding signal that is accessible to all agents during training. This assumption can be realistic for certain scenarios such as agents learning to communicate to play soccer (all agents can perceive the score of the game). However, it is not for other conditions such as teaching where agents have asymmetrical affordances and knowledge, and where communication is a means for a more knowledgeable agent (teacher) to guide a less knowledgeable agent (student) towards the goal. Fig. 3.10 illustrates how the ABP differs from MARL communication and IRL setups.

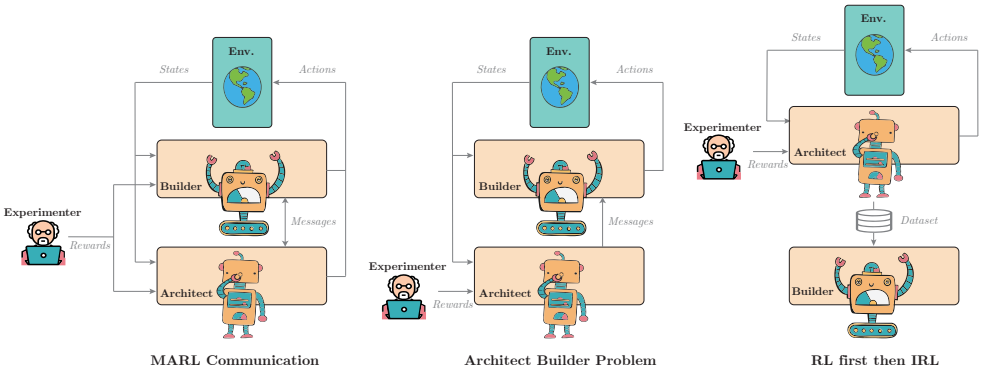


Figure 3.10: The Architect Builder Problem and how it differs with respect to other AI paradigms. Conversely to MARL communication (a), in ABP, the architect cannot act in the environment and the builder never perceives the reward (b). Because the architect cannot act in the environment, it is impossible to frame the problem as an RL and then IRL problem (c).

The ABP is in fact a computational implementation of an experimental semiotics investigation: the Coconstruction Game (Vollmer et al., 2014). In their experiment, the builder and the architect are humans. They are located in separate rooms. The architecture has a picture of a target lego block structure while the builder is seated at a table in front of a set of lego blocks. The architect monitors the builder workspace via a camera (video stream) and must send messages to the builder until it manages to construct the structure. In order to prevent pre-existing communication systems from influencing the results of their studies, the architect uses a button box with neutral symbols (designed to minimize the presence

of biases such as color or shape, so as to avoid the attribution of preexisting meanings). We explore the ABP in chapter 5. More specifically, we propose an algorithmic solution to it in a construction environment like the Coconstruction Game. We investigate the key learning dynamics in terms of mutual information between messages and actions and show that agents can agree on a communication protocol enabling them to generalize to new constructions never seen during training.

3.3 Self-organisation of Trajectories: the Open-ended Skill Acquisition Problem

3.3.1 Computational Models of the Formation of Skill Repertoires with Autotelic RL

Beyond modeling Language formation, developmental AI aims to model how children learn skills in general. In this study, we propose a computational framework that addresses the challenge of self-organizing developmental trajectories and the open-ended learning of skill repertoires. The framework, referred to as autotelic RL or developmental RL, is a combination of developmental approaches and reinforcement learning (see the definition of autotelic in Sec. 1.1.1). It builds on *intrinsic motivations* (IMs) to enable agents to learn to represent, generate, select, and solve their own problems. To provide a comprehensive understanding of the framework, we first present a typology of intrinsic motivation approaches in developmental AI, followed by a presentation of the autotelic learning problem and its solution with autotelic agents.

Intrinsic Motivations in Developmental AI

Developmental AI 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 is 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, 1966; Gopnik et al., 1999; Kidd & Hayden, 2015a; Oudeyer & Smith, 2016; Gottlieb & Oudeyer, 2018).

The integration of IMs into artificial agents thus seems to be a key step towards autonomous learning agents (Schmidhuber, 1991; 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 seminal approaches leverage *population-based* optimization algorithms, i.e. non-parametric models trained on (outcome, policy) pairs. These methods train separate policies for each goal, often demonstrate limited generalization capabilities, and cannot easily handle high-dimensional perceptual spaces.

Recently, we have been observing a convergence between developmental robotics and deep RL, forming a new domain that we propose to call *developmental reinforcement learning* as a subfield of developmental AI. Indeed, RL researchers now incorporate fundamental ideas from the developmental robotics literature in their own algorithms, and reversely developmental robotics learning architectures 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, 1991; 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., 2019; Blaes et al., 2019; Pong et al., 2020).

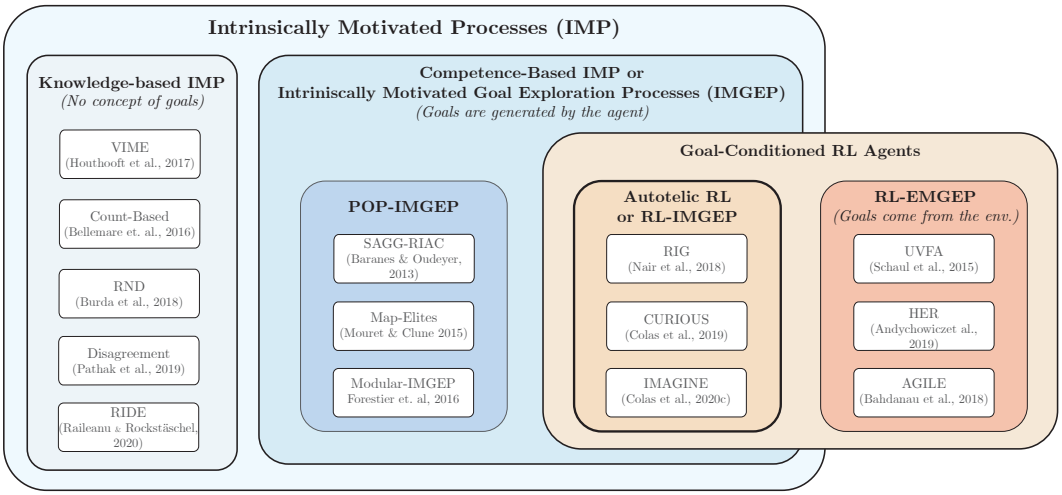


Figure 3.11: A typology of intrinsically-motivated and/or goal-conditioned RL approaches. POP-IMGEP, RL-IMGEP and RL-EMGEP refer to population-based intrinsically motivated goal exploration processes, RL-based IMGEP and RL-based externally motivated goal exploration processes respectively.

Figure 3.11 proposes a visual representation of intrinsic motivations approaches (knowledge-based IMs vs competence-based IMs or IMGEPs) and RL approaches (intrinsically vs externally motivated). RL algorithms using *knowledge-based* IMs (on the left) leverage ideas from developmental robotics to solve standard RL problems. On the other hand, algorithms using competence-based IMs organize exploration around self-generated goals and can be seen as targeting a developmental robotics problem: the *open-ended formation of skill repertoires*. *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). The intersection between IMGEP and multi-goal RL are autotelic RL algorithms or RL-IMGEP. They train agents to generate and pursue their own goals by training goal-conditioned policies. They contrast with RL-EMGEP agents which do not generate their own goals and rely on externally provided ones.

The Autotelic Learning problem

In the *autotelic learning problem* or the *open-ended formation of skill repertoires*, the agent is set in an open-ended environment without any pre-defined goal and needs to acquire a repertoire of skills. Here, we use the definition of skill provided in Sec. 2.3, i.e. 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. Fig. 3.12 illustrates the key difference between multi-goal RL (Sec. 2.3) and autotelic RL. In multi-goal RL an experimenter provides goals and rewards to the agent.

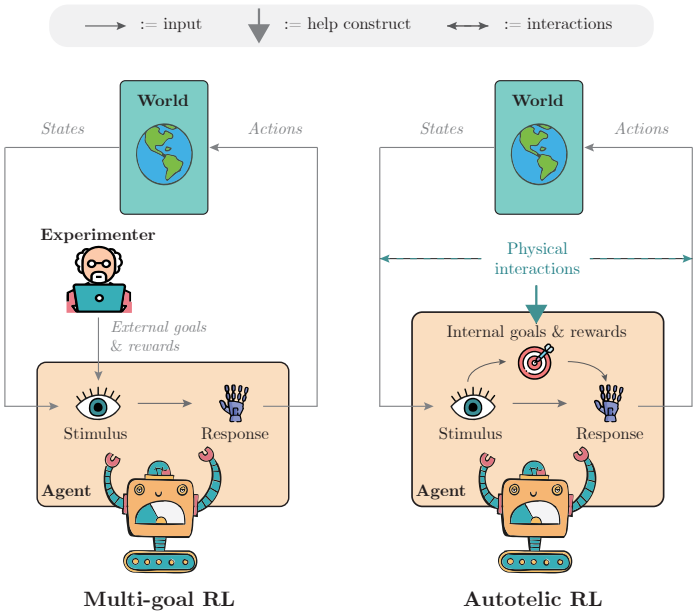


Figure 3.12: Multi-goal RL vs Autotelic RL: In autotelic RL, agents learn to represent, generate, pursue and master their own goals. Goals are thus internal to the agent while multi-goal RL relies on an external experimenter providing goals and rewards.

Evaluating Autotelic 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., 2020b). 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).

- **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 *interesting* interactions defined subjectively by the experimenter (interactions with objects as we do in chapter **??**).
- **Measuring generalization:** The experimenter can 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., 2020b).
- **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., 2019; Blaes et al., 2019; Pong et al., 2020; Akakzia et al., 2021).
- **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.

Autotelic RL Agents

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 autotelic learning problem. 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 $\mathcal{R}_\mathcal{G}$. This four modules are illustrated in Fig. 3.13. In practice, only a few architectures tackle the four learning problems above. Indeed, 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.*

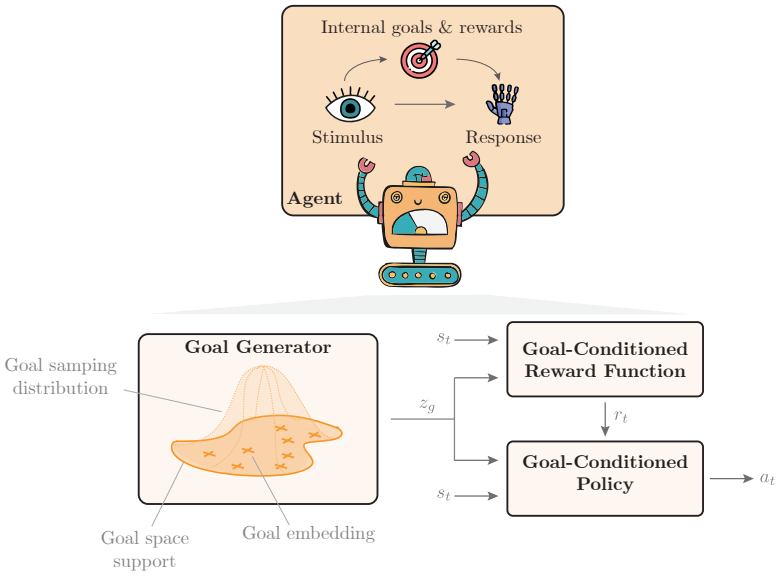


Figure 3.13: Representation of the different learning modules in an autotelic agent.

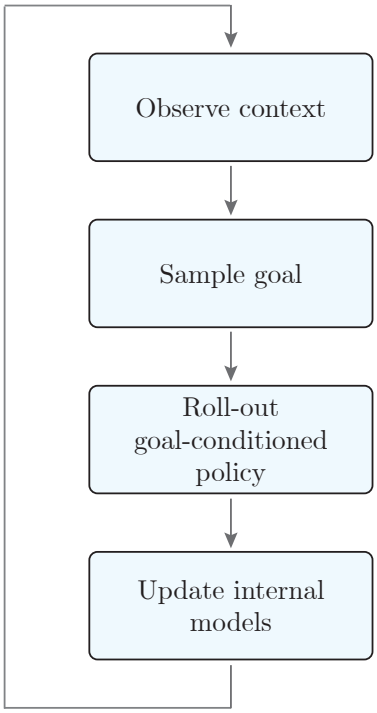
Algorithm 2 details the pseudo-code of RL-IMGEP algorithms. Starting from randomly initialized modules and memory, RL-IMGEP agents enter a standard RL interaction loop. They first observe the context (initial state), then sample a goal from their goal sampling policy. Then starts the proper interaction. 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.

Algorithm 2: 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 $\Pi_{\mathcal{G}}, R_{\mathcal{G}}, \mathcal{Z}_{\mathcal{G}}$



General RL-IMGEP loop

Most RL-EMGEP approaches use pre-defined goal representations where goal spaces and associated rewards are pre-defined by the engineer and are part of the task definition (see our topology of goal representation in Sec. 2.3). On the other hand, 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 Fig. 3.13. In addition to constructing a goal space, autotelic agents must sample goals within that space to actually explore the world. The next two sections address the questions of how to learn goal representations and how to select goals.

How to Learn Goal Representations?

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., 2019a; Jiang et al., 2019; Bahdanau et al., 2019c; Hill et al., 2020; Cideron et al., 2020; 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 reward functions. A few approaches go even further and learn their own goal-conditioned reward function. 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. This method is reminiscent of *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.

In the domain of language goals, 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.

Learning the supports of goal distributions. Finally, to represent collections of goals, agents need to represent the support of the goal distribution — which embeddings correspond to valid goals and which do not. To this end, 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., 2015; Andrychowicz et al., 2017; Nair et al., 2018a; Plappert et al., 2018; Colas et al., 2019; 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. However, some approaches use the set of previously experienced representations to form the support of the goal distribution (Veeriah et al., 2018; Akakzia et al., 2021; 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.

How to Select Goals?

Once autotelic agents have constructed a goal support inside a goal space, they need to specify a goal selection policy. Although agents can sample their goal space uniformly, informed goal selection can be a way for agents to organize their learning curriculum automatically.

Automatic curriculum learning (acl). Applied for goal selection, ACL is a mechanism that organizes goal sampling so as to maximize long-term performance improvement (distal objective). As this objective is usually not directly differentiable, curricu-

lum learning techniques usually rely on a proximal objective. Proxies include *intermediate difficulty* (Sukhbaatar et al., 2018; Campero et al., 2021; Zhang et al., 2020), *novelty-diversity* (Warde-Farley et al., 2019; Pong et al., 2020; Pitis et al., 2020; Kova et al., 2020; Fang et al., 2021) or *medium-term learning progress* (Baranes & Oudeyer, 2013; Moulin-Frier et al., 2014; Forestier & Oudeyer, 2016; Fournier et al., 2018, 2021; Colas et al., 2019; Blaes et al., 2019; Portelas et al., 2020a). Interested readers can refer to Portelas et al. (2020b), which present a broader review of ACL methods.

Hierarchical reinforcement learning (hrl). HRL can be used to guide the sequencing of goals (Dayan & Hinton, 1993; Sutton et al., 1998, 1999; Precup, 2000). 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).

Summary

In this section, we presented the autotelic RL framework. This paradigm, at the intersection of developmental robotics and standard AI technics, builds intrinsically motivated agents that generate and pursue their own problems. Autotelic agents fall in the category of competence-based IMs. Unlike standard multi-goal RL agents that rely on externally provided goals, autotelic agents discover and learn to represent their own goals from their experience of the physical world. This ability to develop in symbiosis with the physical world is reminiscent of Piaget's developmental psychology (Piaget, 1952) which highlights children's ability to shape their learning trajectories with respect to their sensory-motor experience of the world.

3.3.2 Problem Definition

In the second part of this manuscript, we will investigate the role of cultural conventions in the self-organization of agents' developmental trajectories. To do so, we will extend the autotelic RL framework presented in the previous section (Sec. 3.3.1). Complementing the Piagetian approach of autotelic RL and inspired by the literature deriving from Vygotsky (1934)'s theory of child development, we propose a new framework called *Vygotskian Autotelic* ai.

The initial contribution of Part II is conceptual. It proposes to draw the contours of an AI framework where agents leverage pre-existing cultural conventions to transform their learning abilities. It is important to note that, in contrast with the first portion of this research, this investigation will examine the scenario of artificial agents using pre-established cultural conventions to organize their developmental trajectories, disregarding any negotiation of protocols or multi-agent dynamics. In Vygotksian Autotelic agents do not only interact with the physical world surrounding them but with social partners. They are immersed in a (rich) sociocultural environment. An illustration of the difference between autotelic RL and Vygotskian autotelic RL is provided in Fig. 3.14. In Sec. **??**,

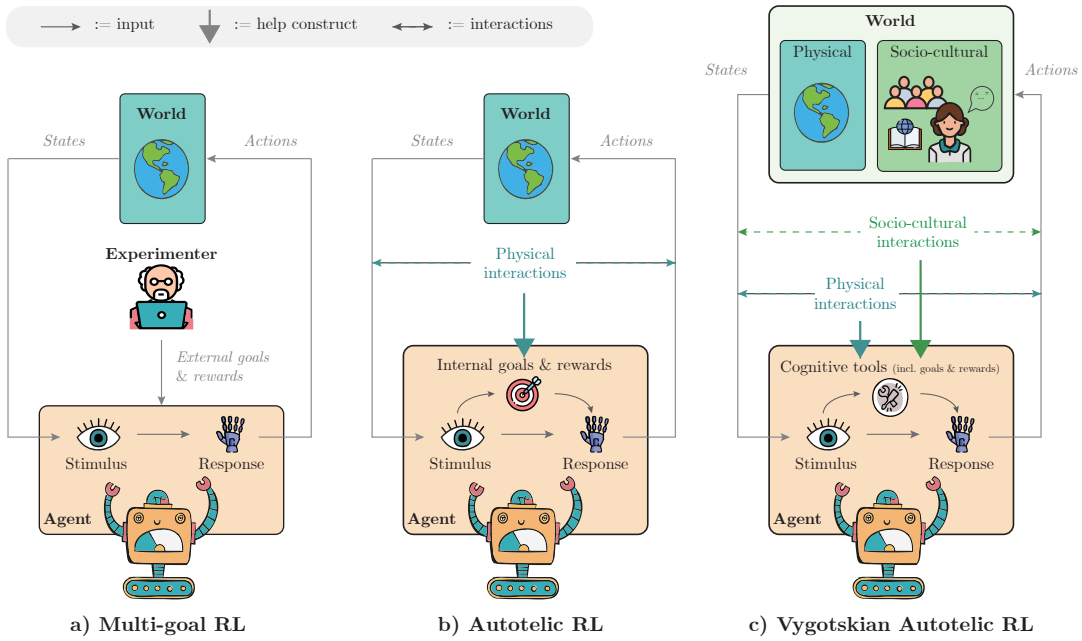


Figure 3.14: From multi-goal RL to autotelic RL to Vygotskian autotelic RL. RL defines an agent experiencing the state of the world as stimuli and acting on that world via actions. Multi-goal RL (a): goals and associated rewards come from pre-engineered functions and are perceived as sensory stimuli by the agent. Autotelic RL (b): agents build internal goal representations from interactions between their intrinsic motivations and their physical experience (Piagetian view). Vygotskian autotelic RL (c): agents internalise physical and socio-cultural interactions into *cognitive tools*. Here, *cognitive tools* refer to any self-generated representation that mediates stimulus and actions: self-generated goals, explanations, descriptions, attentional biases, visual aids, mnemonic tricks, etc.

To benefit from sociocultural and linguistic conventions, Vygotksian autotelic agents need to ground them into their own sensory-motor modalities. They need to extract the structure of language and align it with their sensory-motor experience. As will be discussed, agents do not merely extract the structure of language, but instead assimilate the entire convention, generating an internal representation of the social partner with whom they are interacting. This internal model can be called at any time to generate plans in an autotelic fashion for instance. We argue that this Vygotskian framework can palliate autotelic agents' serious limitations in terms of goal diversity, exploration, generalization, or skill composition. To back this claim, we present two experimental contributions displaying how agents can ground complex spatiotemporal language (Sec. **??**) and how they can use language as a cognitive tool to generate goals in curiosity-driven exploration (Sec. **??**). Both of these experimental contributions will leverage the *Playground* environment: a socio-physical environment made of a variety of objects with different properties and a simulated social partner providing linguistic descriptions of interesting interactions. **Todo:***give more details on experimental contribs*

Part I

Formation of Cultural Conventions

Chapter 4

Self-Organization of a Sensory-motor Graphical Language

Contents

In this chapter, we investigate whether artificial agents can develop a shared language in an ecological setting where communication relies on a *sensory-motor channel*. To this end, we extend the setup of neural language games described in Sec. 3.2.1 and introduce the Graphical Referential Game (GREG). In the GREG, 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 demonstrate that CURVES not only succeed at solving the GREG but also enable agents to self-organize a language that generalizes to feature compositions never seen during training. In addition to evaluating the communication performance of our approach, we also explore the structure of the emerging language. Specifically, we show that the resulting language forms a coherent lexicon that is shared between agents and that basic compositional rules on the graphical productions could not explain the compositional generalization

4.1 Motivations

As we described in Sec. 3.2.1, most approaches to language games 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*.

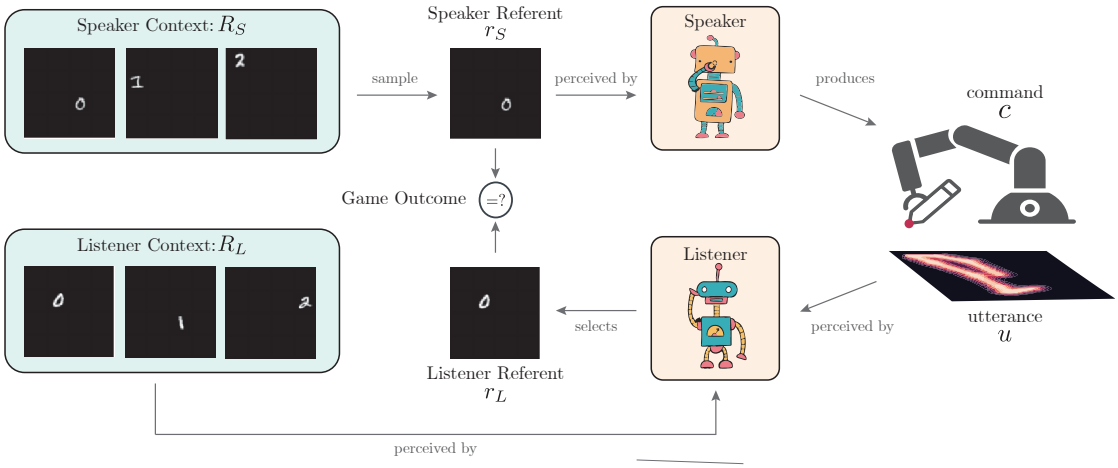


Figure 4.1: **The Graphical Referential Game:** During an instantiation of the game, the speaker's goal is to produce a motor command c that will yield an utterance u in order to denote a referent r_S sampled from a context $\tilde{R}_S$. Following this step, the listener needs to interpret the utterance 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 ($r_L \equiv r_S$).

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 chapter, 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 Fig. 4.1) has to produce a graphical *utterance* given a single target *referent* while a *Listener* agent (bottom of Fig. 4.1) 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 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 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. They, moreover, allow gradients to back-propagate from listener to speaker while we

use a decentralized approach. 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 contribution 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.

Specific Contributions

The specific contributions introduced in this chapter are:

- The Graphical Referential Game (GREG): a variation of the referential language 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 study of CURVES's generalization performances on compositions of features never seen during training in a simplified control setting and a more perceptually challenging one.
- A complementary analysis of the structure of the emerging graphical language measuring lexicon coherence and compositionality scores derived from the Haussdorf distance.

4.2 The Graphical Referential Games

We consider a group of two agents playing a fixed number of referential games, each time alternating their roles (speaker or listener). During a game, we first present a context R of n objects, 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 $(\hat{r})$ matches the target $r^\star$.

The setup

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: $\mathcal{R}_m^k = \{\sum_{f \in S} f | S \subseteq F_m, |S| = k\}$. 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. 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. More precisely, the speaker perceives the context R as $\tilde{R}_S$ and its target $r^\star$ as $r^\star_S$. Similarly, the listener perceives the context R as $\tilde{R}_L$ and selects a referent $\hat{r}$ among it.

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 $\tilde{R}_S \neq \tilde{R}_L$.

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 Fig. 4.2(a). The arm motion is derived from Dynamical Motor Primitives (DMPs) (Schaal, 2006). The DMP is parametrized by a command vector $c \in \mathbb{R}^{20}$. Each of the x and y positions of the pen is controlled by a DMP starting at the center of the image and parameterized by 10 weights. These weights are the parameters of the motion of a one-dimensional oscillator that generates a smooth drawing trajectory T made of 10 coordinates $T = \{v_i\}_{i=0,\dots,9}$. The parameters of the two DMPs are given in Suppl. table A.1. The trajectory is then fed to a Differentiable Sketching model (Mihai & Hare, 2021a) generating an $D \times D$ image (in our implementation, $D = 52$).

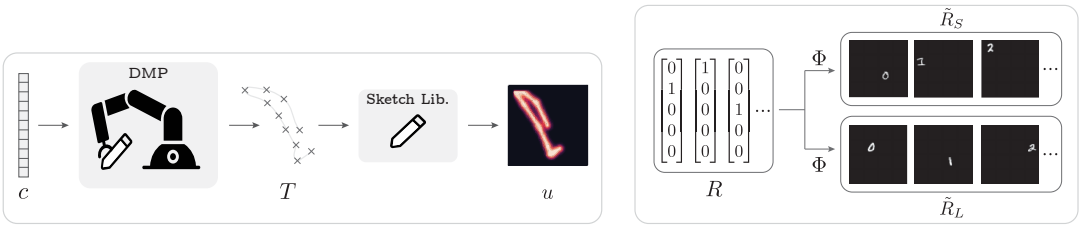


Figure 4.2: (a) **Sketching sensory-motor system**: The sensory-motor system imitates a robotic arm drawing a sketch on a 2D plan. DMPs first convert a continuous command c into a sequence of coordinates T. This trajectory is then rendered as a 52×52 graphical utterance thanks to a differentiable sketching 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.

Objectives

In this study, we aim to answer the three following questions:

1. What are agents' communicative performances in the GREG? Are agents able to solve the game? Are they able to generalize to compositional referents?
2. Are the emergent signs coherent? Do agents produce the same utterances to denote the same referents?
3. Are the emergent signs compositional? Are there compositional rules in the production of signs naming compositional referents? [1]

Are agents able to solve the greg*?* To answer the first question, we will monitor the communicative performance of agents on both training and testing referents. The training referents consist of a single feature: $\mathcal{R}_{\text{train}} = \mathcal{R}_5^1$ while the testing referents consists of two features: $\mathcal{R}_{\text{test}} = \mathcal{R}_5^2$. For visual examples of compositional referents, see Suppl. Section A.1.2.

Are the emergent signs coherent? To measure coherence we propose to use a similarity measure based on the Hausdorff distance. Haussdorf distance is known to capture geometric features of trajectories, in particular, their shape (Besse et al., 2015). 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.

- Agent 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
- Referent Coherence (R-coherence): For a given agent, measure the mean pairwise similarity between utterances produced for different referents.

Are the emergent signs compositional? To measure the compositionally of the utterances, we introduce a topographic score based on the Hausdorff distance ρ. ρ quantifies how an utterance denoting a compositional referent made of feature i and j ($u(r_{ij})$) is actually closer to the 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 A.1.3.t

4.3 CURVES: Contrastive Utterance-Referent associatiVE Scoring

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

1. The contrastive learning of an energy landscape $E(r, u)$, defined as the cosine similarity between utterance and referent embeddings.

[1] Note that the ability to perform compositional generalization (question 1) and the presence of compositional structure in utterances (question 3) are two separate investigations.

2. The generation of an utterance that maximizes the energy for a given target referent $r_S^\star$.

Agents modules and interactions.

Each agent $A \in \{A_1, A_2\}$ perceives utterances and referents using two distinct CNN encoders f_A (for referents) and g_A (for utterances)[2]. f_A and g_A map referents and utterances 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 Fig. 4.3(a). The agent then computes the energy landscape as: $E_A(r, u) = \cos(f_A(r), g_A(u))$.

A given referential game unfolds as follows. Agents have 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 approach the utterance $u^\star$ that maximizes $E_S(r_S^\star, u)$. The listener observes $\hat{u}$ and selects referent $\hat{r}$ in context $\tilde{R}_L$ that maximizes $E_L = (r, \hat{u})$:

$$\begin{cases} \hat{u} \approx u^\star = \underset{u \in \mathcal{U}}{\operatorname{argmax}}\ E_S(r_S^\star, u) \\ \hat{r} = \underset{r \in \tilde{R}_L}{\operatorname{argmax}}\ E_L(r, \hat{u}) \end{cases} \tag{4.1}$$

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.

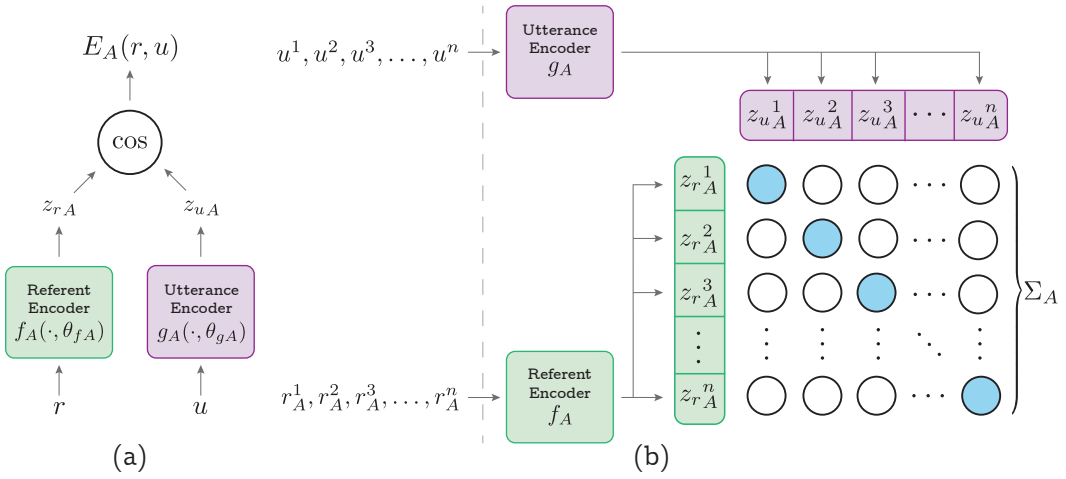


Figure 4.3: (a) **Agents's dual encoder architecture.** Referents and utterances are mapped to a share latent space. The energy between a referent r and an utterance u is computed as the cosine similarity between their respective embeddings. (b) **Cosine similarity matrix update from collected samples.** Agents compute the energy for all referents and utterances they collected to form the squared matrix Σ_A. During contrastive updates agents maximize blue circles and minimize white ones.

[2] when referents are one-hot vectors f_A is a fully-connected network. Parameters for both encoders are given in Suppl. table A.2.

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 $\{(r_S^i, u^i, o^i)\}_{i=1,\dots,n}$ while the listeners observes $\{(u^i, r_L^i)\}_{i=1,\dots,n}$. 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 Fig. 4.3(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.

$$J_A(\Sigma_A, i) = \frac{CE((\Sigma_A)_{i,1:n}, e_i) + CE((\Sigma_A)_{1:n,i}, e_i)}{2} \tag{4.2}$$

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):

$$\underset{\theta_{f_S}, \theta_{g_S}}{\text{minimize}} \sum_{i=1}^{n} o_i J_S(\Sigma_S, i) \tag{4.3}$$

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

$$\underset{\theta_{f_L}, \theta_{g_L}}{\text{minimize}} \sum_{i=1}^{n} J_L(\Sigma_L, i) \tag{4.4}$$

Note that in Eq. 4.4, r_L^i is the target referent perceived by the listener. This means that, at the end of the game, the speaker indicates the referent (as perceived by the listener) that they named. As reviewed in Sec. 3.3.1, this retroactive pointing mechanism was employed in both early language game implementations (Steels, 1995a) and more recent ones (Lazaridou et al., 2017; Chaabouni et al., 2020; Portelance et al., 2021).

Speaker's utterance optimization.

We distinguish two utterance generation strategies:

- The descriptive generation: in which the speaker agent only considers the target referent $r_S^\star$ to produce an utterance that maximizes the cosine similarity between the embeddings of $r_S^\star$ 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 4.1 and seek for the optimal motor command $c^\star$ using gradient ascent:

$$c^\star = \underset{c \in \mathbb{R}^p}{\text{argmax}}\ E(r_S^\star, M(c)) \tag{4.5}$$

- The discriminative generation: in which 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 $\tilde{R}_S$:

$$c^\star = \underset{c \in \mathbb{R}^p}{\text{argmin}}\ CE(\sigma_S, e_{r_S^\star}) \tag{4.6}$$

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.

4.4 Experiments and Results

4.4.1 Communicative Performance

In all three settings of the Graphical Referential Game (one-hot, visual-shared, and visual-unshared), agents succeed and achieve a perfect training success rate of 1.

Generalization to compositional referents.

Table 4.1 exposes the generalization performances 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 ($\text{SR}_{\text{1-feat}} = 0.25$).

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

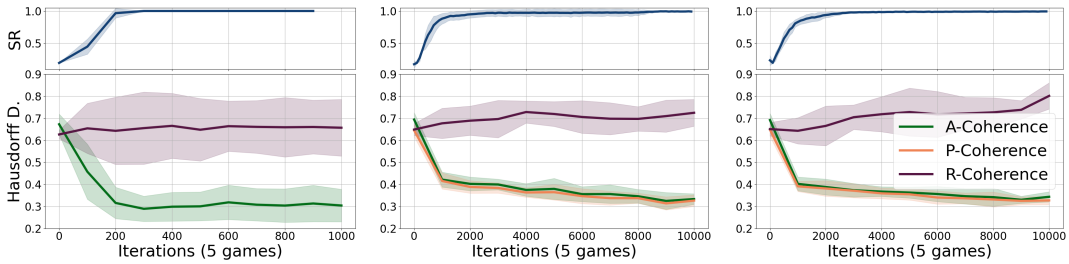
Table 4.1: **Generalization performances.** Success rates evaluated on exhaustive context $|R| = 10$ with referents $r \in \mathcal{R}_5^2$ for both generative (Eq. 4.5) and discriminative (Eq.4.6) utterance generation.

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 A.2.1 seem to corroborate this hypothesis. Surprisingly, we observe that success rates for descriptive (Eq. 4.5) and discriminative (Eq.4.6) 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.

4.4.2 Structure of the Emergent Language

Coherence

Fig. 4.4 displays the evolution of the inter-agent (A), inter-perspective (P), and inter-referent (R) coherence during training. A group starts to converge and succeed at the game 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. 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 4.4: **Training success rate (SR) and Coherence distances** (a) one-hot referents (b) visual-shared referents (c) visual-unshared referents.

As displayed in Fig. 4.5, the language used by agents self-organizes around five distinct symbols. It is important to note that this self-organization arises from the production of continuous signals with no explicit communication of the five categories of visual referents. Other visualizations for one-hot and shared visual referents are available in Suppl. Section A.2.2. We also provide illustrations of P-coherence in Suppl. Section A.2.3.

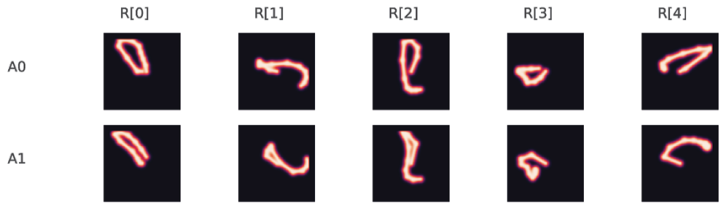
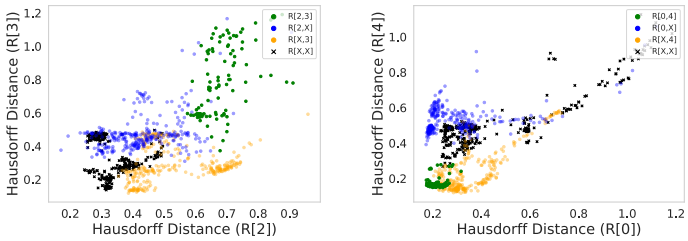


Figure 4.5: **Instance of an emerging lexicon.** Utterances are produced by a pair of agents trained with unshared perspectives (1 seed). The perspective for each referent is chosen randomly.

Compositionality

In Section 4.4.1, we showed that agents achieve a near-perfect success rate at naming compositions of one-hot features at test time. Is this successful communication reflected by a compositional structure in the produced signs? To investigate this question we propose the topographic maps associated with their topographic scores in Fig. 4.6.



(a) $\rho = -0.401$ (b) $\rho = 0.147$

Figure 4.6: **Topographic map examples for a single seed in one-hot referents setting**. Each utterance names a compositional referent and is colored in blue if it contains feature i ($R[i, X]$), orange if it contains feature j ($R[X, j]$), green if it contains both ($R[i, j]$), and black if it contains none ($R[X, X]$). (a) Corresponding to the worst topographic score $\rho = -0.401$ (combination of feature $i = 2$ and $j = 3$) (b) Corresponding to the best topographic score $\rho = 0.147$ (combination of feature $i = 0$ and $j = 4$).

Each point in a topographic map is an utterance naming a compositional referent $r \in \mathcal{R}_5^2$ and has coordinate $(d_H(u(r_i), \cdot), d_H(u(r_j), \cdot))$. Utterances at the bottom left of the topographic maps are therefore simultaneously close to the two utterances naming the isolated features. All the topographic maps are available in Fig. A.10 of Suppl. Section A.2.4. They show that for a minority of compositions (3 out of 10), the utterances naming the composition of two features are not close in Haussdorf distance to the utterances naming the two isolated features ($\rho < 0$). This indicates that proximity in Haussdorf distance is not a necessary condition for agents to generalize on compositional referents. The matrix of composition provided in Fig. 4.7 illustrate that it is indeed very difficult to infer a composition rule from the generated utterances.

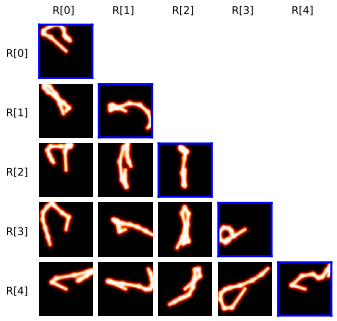


Figure 4.7: **Matrix of compositions.** Blue frames represent utterances generated for a perspective in $\mathcal{R}_5^1$, other utterance denote the corresponding compositions in $\mathcal{R}_5^2$

Despite the fact that we cannot perceive the compositional structure of emerging signs, the internal representations of agents seem to leverage compositional mechanisms. The t-snes provided in Fig. 4.8 shows that the embeddings for both compositional referents and the utterances naming them are close to their constituents.

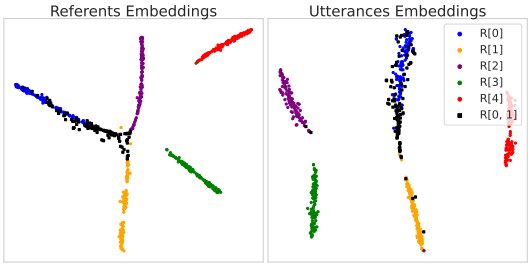


Figure 4.8: **T-sne of utterance and referent embeddings.** Embeddings are computed for 100 perspectives in the visual-unshared setting. Additional t-snes are provided in Suppl. Section A.2.6.

Conclusion

If the Haussdorf distance does not enable us to identify compositional rules in the production of utterances, it is particularly relevant for describing their coherence. This paper, therefore, provides the first step toward understanding the mechanisms at hand for the emergence of structure in self-organizing languages. The structural analysis we present sheds light on the importance of studying ecological systems. Indeed, agents directly optimizing utterances in pixel space can negotiate a successful communication protocol (as indicated in table 4.2) but the absence of structure in the resulting lexicon (illustrated in Fig. 4.9) prevents us from analyzing the properties of utterances.

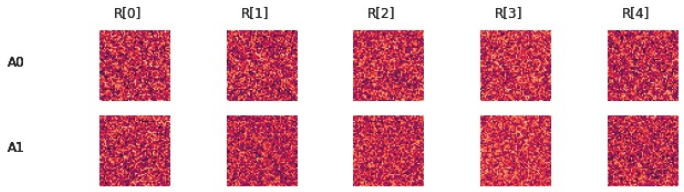


Figure 4.9: **Emerging lexicon without motion primitives.** Utterances naming referents with unshared perspectives.

	$\mathbf{SR}_{\text{train}}$	$\mathbf{SR}_{\text{test}}$
One-hot	0.99 ± 0.01	0.96 ± 0.02
Visual-shared	0.99 ± 0.01	0.55 ± 0.03
Visual-unshared	0.99 ± 0.01	0.41 ± 0.02

Table 4.2: **Training and generalization success without DMPs.** Utterances are generated in descriptive mode, and visual referents are seen from different perspectives.

4.5 Discussion and Future Work

In this chapter we formalized GREG: a new ecological 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. The representations that agents learn enable them to communicate about compositional referents never encountered during training. If the Haussdorf distance illustrates that emergent signs are coherent, it does not capture compositionality among them.

Future work may leverage our ecological setup and algorithmic solution to experiment with and test a variety of hypotheses that influence structures in self-organizing sing systems. An analysis of the impact of the sensory-motor constraints on the topology of graphical signs could for instance provide valuable insight into the ecological factors facilitating the emergence of a compositional graphical language. Inspired by work on the cultural evolution of language (Kirby, 2001), our setup can also serve as a basis to investigate and visualize the impact of other factors such as population dynamic or cognitive abilities of agents (with varying memory or perceptual systems). Finally, 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 optimize 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).

Chapter 5

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

Contents

In contrast to the preceding chapter, which examines the self-organization of cultural conventions in the context of sensory-motor constraints in the classical language (or referential) game, the present chapter proposes to investigate the emergence of goal-directed communication between artificial agents in a novel setting. More specifically, we study the collaboration between 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. This setting fundamentally differs from the standard MARL communication setup (presented at the end of Sec. 3.2.1) in which the reward function is provided to all agents.

In this new setting, the agents need 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. Experimental Semiotics research has demonstrated human proficiency in learning from a priori unknown instructions and meanings. This study draws inspiration from Experimental Semiotics and introduces

the Architect-Builder Problem (ABP). In this asymmetrical setting, an architect must learn to guide a builder toward 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.

5.1 Motivations

Humans have a remarkable ability to teach and learn from each other, which allows knowledge and skills to be shared and refined across generations. Even in situations where there is no shared language or common ground, such as a parent teaching a baby how to stack blocks during play, people can teach and be taught. 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 signals, demonstrations, or shared communication protocols. 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 made of Lego blocks. 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. Are artificial agents capable of developing such cultural conventions?

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 RL (Sec. 2.1) and even MARL (Sec. 2.4) where explicit rewards are available to all agents. (2) implies the absence of teleoperation or third-person demonstrations and thus distinguishes the ABP from IL (Sec. 2.2). Finally, (3) prevents the architect from relying on a fixed communication protocol since the meanings of instructions must be negotiated. Artificial agents exploiting pre-defined cultural conventions will be explored in part II of this manuscript.

These three 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 interactions.

With all these constraints in mind, we propose Architect Builder Iterated Guiding (ABIG), an algorithmic solution to ABP where both agents are artificial agents. 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 toward 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, meaning that when one agent learns, the other agents behavior is fixed. During the first frame (the modeling 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.

Specific Contributions

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).

5.2 The Architect-Builder Problem

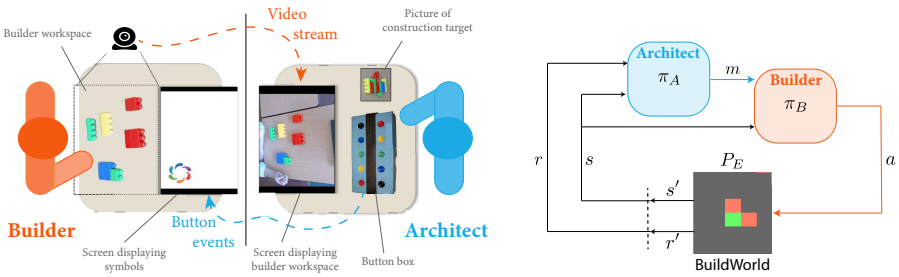
The Architect-Builder Problem. We consider a multi-agent setup composed of two agents: an architect and a builder. Both agents observe the environment state s but only the architect knows the goal at hand. The architect cannot take actions in the environment but receives the environmental reward r whereas 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 Fig. 5.1(a) while the overall architect-builder-environment interaction diagram is given in Fig. 5.1(b). The differences between the ABP setting and the MARL and IRL settings are illustrated in Fig. B.2.

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. 'H-Line': The agent must place all the blocks in a horizontal line configuration;
4. 'V-Line':The agent must place all the blocks in a 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.



(a) (b)

Figure 5.1: (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 Fig. B.1(b).

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).

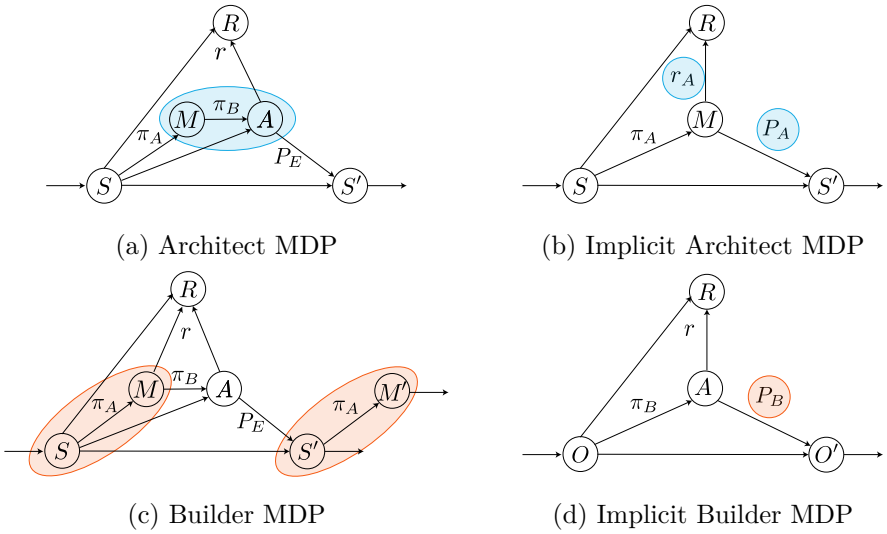
Communication. The architect guides the builder by sending messages m which are one-hot vectors of size $|\mathcal{V}|$ ranging from 2 to 72, see 5.4.5 for the impact of this parameter.

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.

5.3 ABIG: Architect-Builder Iterated Guiding

5.3.1 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 5.2). From the point of view of the architect, messages are actions that influence the next state as well as the reward (see Fig. 5.2 (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



(a) Architect MDP (b) Implicit Architect MDP

(c) Builder MDP (d) Implicit Builder MDP

Figure 5.2: **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.

builder's actions that drive the reward and the next states (see Fig. 5.2 (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 Fig. 5.2 (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 B.1).

$$\left.\begin{aligned} P_A(s'|s,m) &= \sum_{a\in\mathcal{A}} \tilde{\pi}_B(a|s,m) P_E(s'|a,s) \\ r_A(s,m) &= \sum_{a\in\mathcal{A}} \tilde{\pi}_B(a|s,m) r(s,a) \end{aligned}\right\} \quad \text{with} \quad \tilde{\pi}_B(a|s,m) \triangleq P(a|s,m) \tag{5.1}$$

$$P_B(s',m'|s,m,a) = \tilde{\pi}_A(m'|s') P_E(s'|s,a) \quad \text{with} \quad \tilde{\pi}_A(m'|s') \triangleq P(m'|s') \tag{5.2}$$

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. (5.1) 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:

$$\pi_A^* = \underset{\pi_A}{\text{argmax}}\, G_A = \underset{\pi_A}{\text{argmax}}\, \mathbb{E}[\sum_t \gamma^t r_{A,t}] \tag{5.3}$$

$\gamma \in [0,1]$ 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:

$$G_B = G_A = \mathbb{E}_\tau[\sum_t \gamma^t r_{A,t}] = \mathbb{E}_\tau[\sum_t \gamma^t \tilde{r}_t] \tag{5.4}$$

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. (5.1), 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.

5.3.2 Practical Algorithm

ABIG iteratively structures the interactions between a builder-architect pair into interaction frames. Each iteration starts with a *modeling 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 Fig. 5.3 and the pseudo-code is reported in Algorithm 3.

Modeling 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 collecting 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. 5.3). 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 $\mathcal{D}_B$.

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

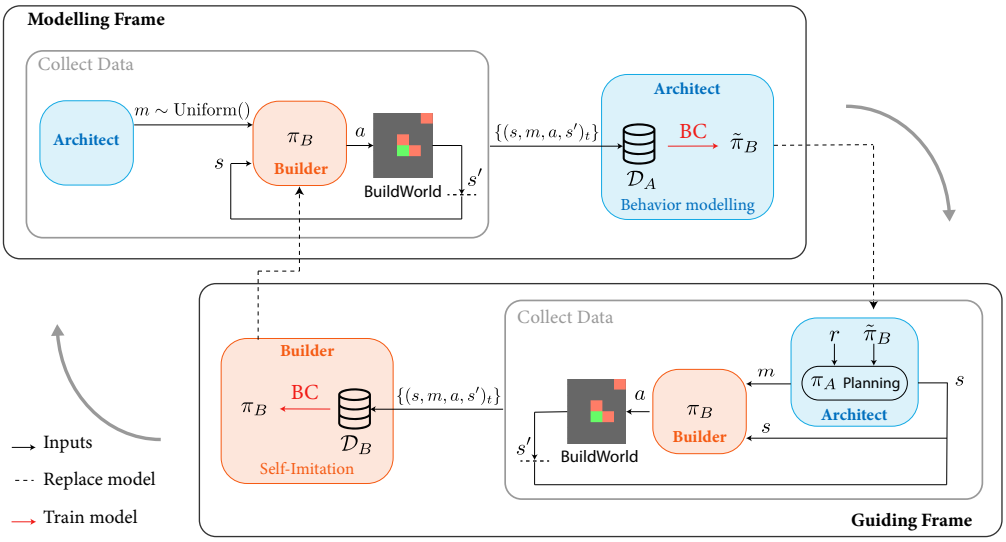


Figure 5.3: **Architect-Builder Iterated Guiding.** Agents iteratively interact through the modeling and guiding frames. In each frame, one agent collects data and improves its policy while the other agent's behavior is fixed.

Algorithm 3: 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

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 B.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.

Control Settings. 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.

5.3.3 Understanding the Learning Dynamics

Intuitive Explanation

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:

$$J(m,\pi) = -\sum_{a\sim\mathcal{D}} \log \pi(a|m) \approx \mathbb{E}_{a\sim\pi(\cdot|m)}[-\log \pi(a|m)] \approx H[\pi(\cdot|m)] \tag{5.5}$$

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 forgetting mechanism to undo unfavorable message-to-action associations. Consequently, in that case, self-imitation tends to simply reinforce the initial builder's preferences over actions making the controllability of the builder policy depend heavily on the initial preferences.

ABIG with a Toy Problem

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 5.4 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 5.4 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 5.5, favorable and unfavorable messages are sampled alike which prevents the forgetting mechanism to undo unfavorable message-to-action associations. Consequently, initial preferences are reinforced.

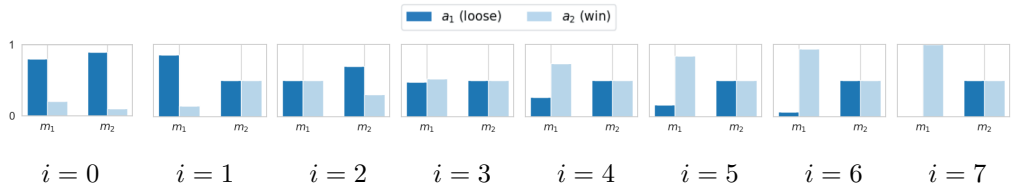


Figure 5.4: ABIG-driven evolution of message-conditioned action probabilities in the toy problem. Initial conditions are unfavorable since a_1 is more likely than a_2 for both messages. ($i = 0$) Given the initial conditions, the architect only sends message m_1 since it is the most likely to result 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.

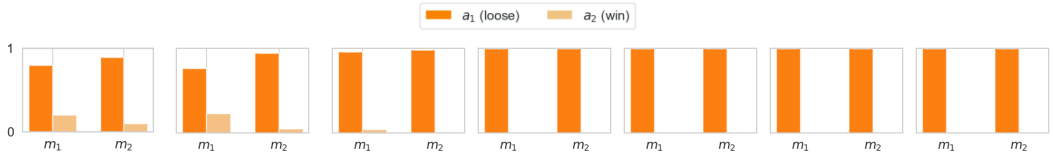
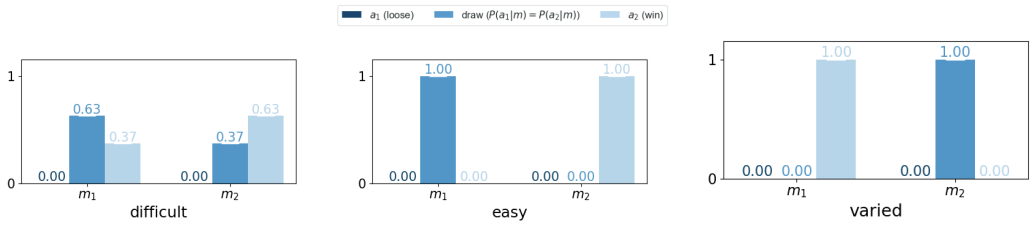


Figure 5.5: ABIG-no-intent driven evolution of message-conditioned action probabilities for a simple problem where builder must learn to produce action a_2. Initial conditions are unfavorable since a_1 is more likely than a_2 for both messages. Without an architect's guiding messages during training, 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.

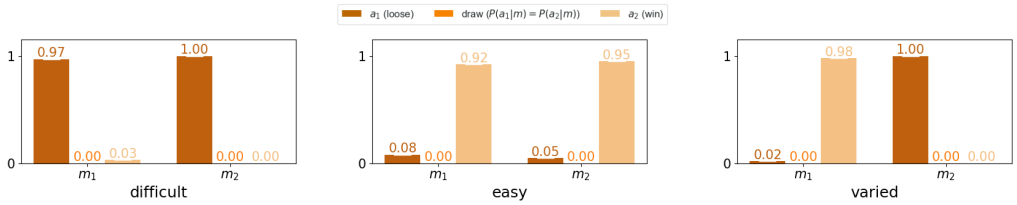
Unfavorable	Favorable	Intermediate
(a) **Initial probabilities**		
$P(a_1\|m_1) = 0.8,\ P(a_2\|m_1) = 0.2$ $P(a_1\|m_2) = 0.9,\ P(a_2\|m_2) = 0.1$	$P(a_1\|m_1) = 0.2,\ P(a_2\|m_1) = 0.8$ $P(a_1\|m_2) = 0.1,\ P(a_2\|m_2) = 0.9$	$P(a_1\|m_1) = 0.9,\ P(a_2\|m_1) = 0.1$ $P(a_1\|m_2) = 0.1,\ P(a_2\|m_2) =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

(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 5.6: **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.

Figure 5.6 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 forgetting mechanism. 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 a high chance 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).

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

5.3.4 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.

IRL has been investigated for HRI when it is challenging to specify a reward function. Instead of defining rewards, IRL 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 a 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 modeling 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.

5.4 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 $|\mathcal{V}| = 18$.

5.4.1 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). Fig. 5.7 reports the mean success rate on the four tasks defined in Section 5.2. 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 5.3.3 that shows that, in favorable settings, a self-imitating builder can develop a reasonably controllable policy (defined in Section 5.3.3) 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.

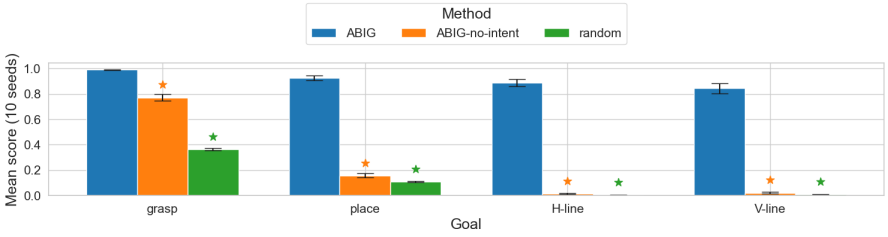


Figure 5.7: Methods performances (stars indicate significance with respect to ABIG model according to Welch's t-test with null hypothesis $\mu_1 = \mu_2$, at level $\alpha = 0.05$). ABIG outperforms control baselines on all goals.

5.4.2 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. Fig. 5.8 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 5.8: 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 Fig. 5.9. 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.

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

Figure 5.9: 6-block-shapes that ABIG can construct in transfer mode when trained on the 'place' task.

5.4.3 Proof of Emerging Language

In this paragraph, we propose to thoroughly study the evolution of the builder's policy in order to provide a deeper analysis of ABIG. Our analysis principally relies on mutual information measures that we define below.

Metric definition. We define three metrics that characterize the builder's 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:*

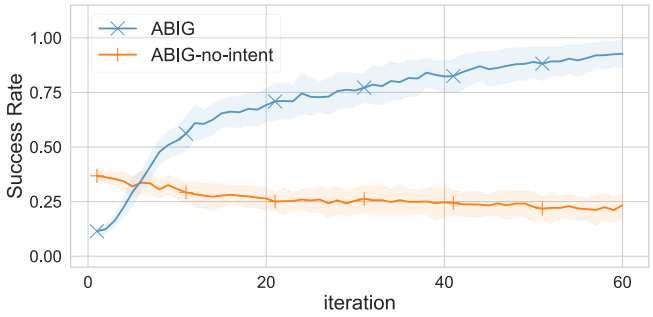
$$\bar{H}(\pi) = \frac{1}{|\mathcal{M}|} \sum_{(s,m) \in \mathcal{M}} \left[- \sum_{a \in \mathcal{A}} \pi(a|s,m) \log \pi(a|s,m) \right]$$

- *Mutual Information between messages and actions*

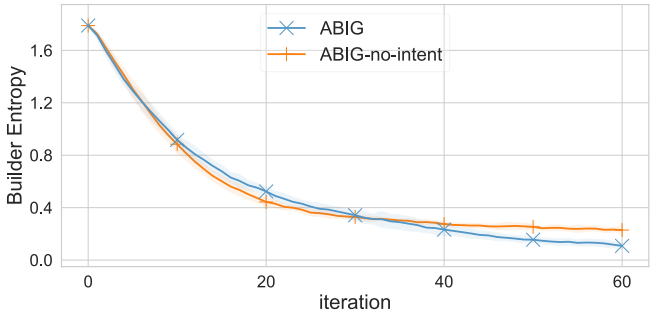
$$I_m = \sum_{m \in \mathcal{V}} \sum_{a \in \mathcal{A}} p_{MA}(m,a) \log \frac{p_{MA}(m,a)}{p_{A}(a) p_{M}(m)}$$

- *Mutual Information between states and actions*

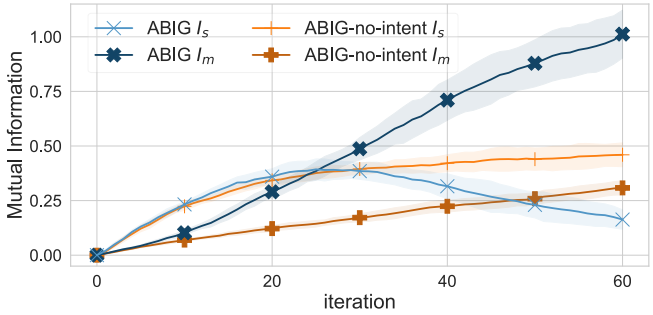
$$I_s = \sum_{s \in \mathcal{M}} \sum_{a \in \mathcal{A}} p_{SA}(s,a) \log \frac{p_{SA}(s,a)}{p_{A}(a) p_{S}(s)}$$



(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 5.10: 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.

Analysis. Figure 5.10 displays the evolution of these metrics after each iteration as well as the evolution of the success rate (a). As indicated by Eq. (5.5), 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.

5.4.4 Additional Baselines

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 5.11, 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 5.11, the initial deterministic policies lack flexibility and fail. This shows that the builder must iteratively evolve its policy in order to make it controllable.

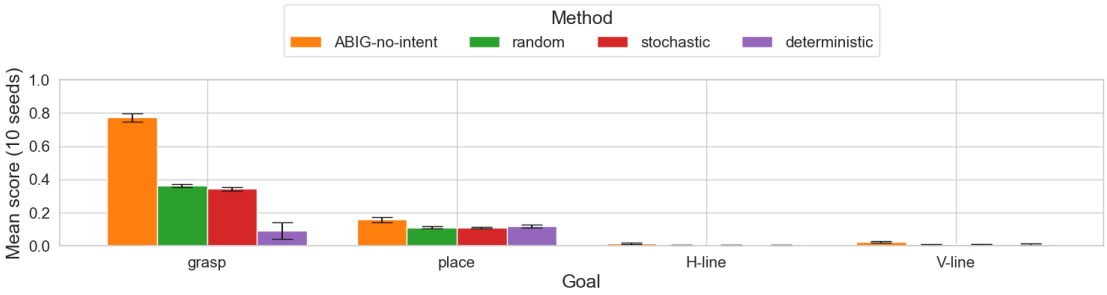


Figure 5.11: Baseline performance depending on the goal: stochastic policy behaves on par with random builder. Self-imitation with ABIG-no-intent remains the most controllable baseline.

5.4.5 Impact of Vocabulary Size

We finally investigate the impact of vocabulary size on ABIG communicative performance in Fig. 5.12. The bigger the vocabulary size, the better the performances suggesting that with more messages available, the architect can more efficiently refer to the desired action.

5.5 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 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

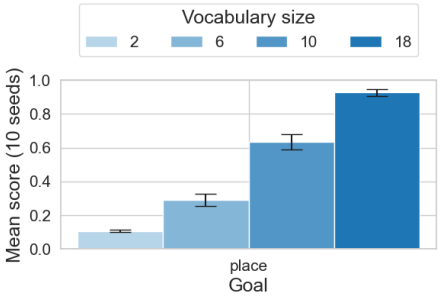


Figure 5.12: Influence of the Vocabulary size for ABIG on the 'place' task. Performance increases with the vocabulary size.

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, 2021), 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.

Part II

Exploitation of Cultural Conventions

Appendices

Appendix A

CURVES

Supplementary Material

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;

A.1 Supplementary Methods

A.1.1 Sensory-Motor System

Dynamic Motion Primitives. This subsection provides additional details about the implementation of the Dynamical Movement Primitives use to produce 2-dimensional trajectories. Our drawing system consists of a 2-dimensional system that mimics the motion of a pen in a plan. Each of the x and y positions of the pen is controlled by a DMP starting at the center of the image and parameterized by 10 weights. These weights are the parameters of the motion of a one-dimensional oscillator that generates a smooth trajectory of 10 points. The parameters of the two DMPs are given in table A.1.

Sketching Library. Trajectories obtained with the DMPs are then mapped to a 52x52 grid which is converted to an image with the `raster` and `softor` functions of the sketching library Mihai & Hare (2021a). The drawing thickness parameter is fixed to $1e-2$.

Parameter	Value
Number of weights	10
Delta time	0.1
Number of points	10
Weights range	[-500, 500]
Position Init.	0

Table A.1: DMP parameters for each of the two coordinate motions

A.1.2 Testing Set

Figure A.1 displays examples of compositional referents made of 2 features.

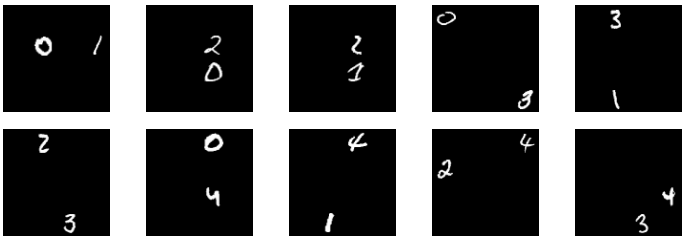


Figure A.1: **Perspective instances of the testing set** $\mathcal{R}_5^2$.

A.1.3 Topographic Score

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

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

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 A.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 A.1.

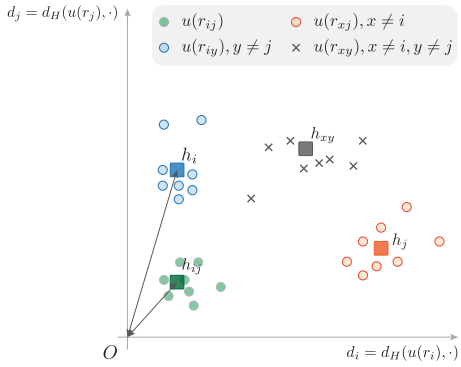


Figure A.2: Idealized mapping of utterances denoting compositional referents in the plan representing distances to utterances naming isolated features i and j.

A.1.4 Training procedure and hyperparameters

Agents have two separate encoders based on the same model architecture described in Table.**??**. 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 A.2: **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 4.5 in the descriptive case or equation 4.6 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 $1e^{-2}$.

A.1.5 Pseudo-code

Algorithm 4: 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**
  $U_S \leftarrow M(c)$
  $Z_u \leftarrow g_S(U)$
  $S \leftarrow \text{sim}_{\cos}(Z_r, Z_u)$
  $\mathcal{L} \leftarrow \text{mean}(\text{diag}(\text{S})) * (-1)$
  GD step on c to minimize $\mathcal{L}$
end for
Return $M(c)$

Algorithm 5: 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 6: 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**
  $\mathcal{L} \leftarrow (\mathcal{L} \cdot o)/N_{\text{referents}}$
else
  $\mathcal{L} \leftarrow (\mathcal{L} \cdot \mathbf{1})/N_{\text{referents}}$
end if
Return $\mathcal{L}$

A.2 Supplementary Results

A.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 A.3: 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 A.4: 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 A.4 & A.3.

A.2.2 Additional Lexicons

Figure A.3: **Instance of an emerging lexicon.** (Utterances are naming visual-shared referents).

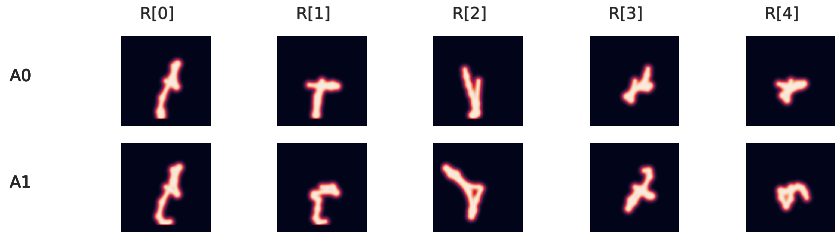


Figure A.4: **Instance of an emerging lexicon.** (Utterances are naming one-hot referents).

A.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 A.5: Utterances examples for referent 0.

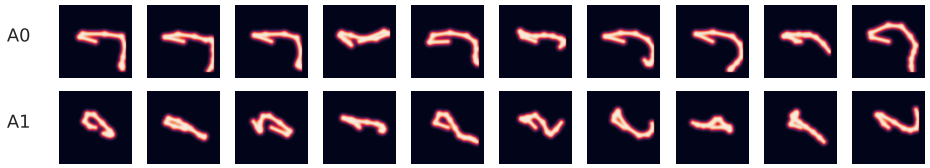


Figure A.6: Utterances examples for referent 1.

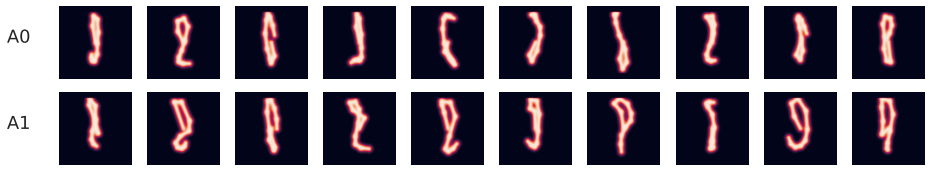


Figure A.7: Utterances examples for referent 2.

Figure A.8: Utterances examples for referent 3.

Figure A.9: Utterances examples for referent 4.

A.2.4 Topographic Maps & Scores

One-Hot

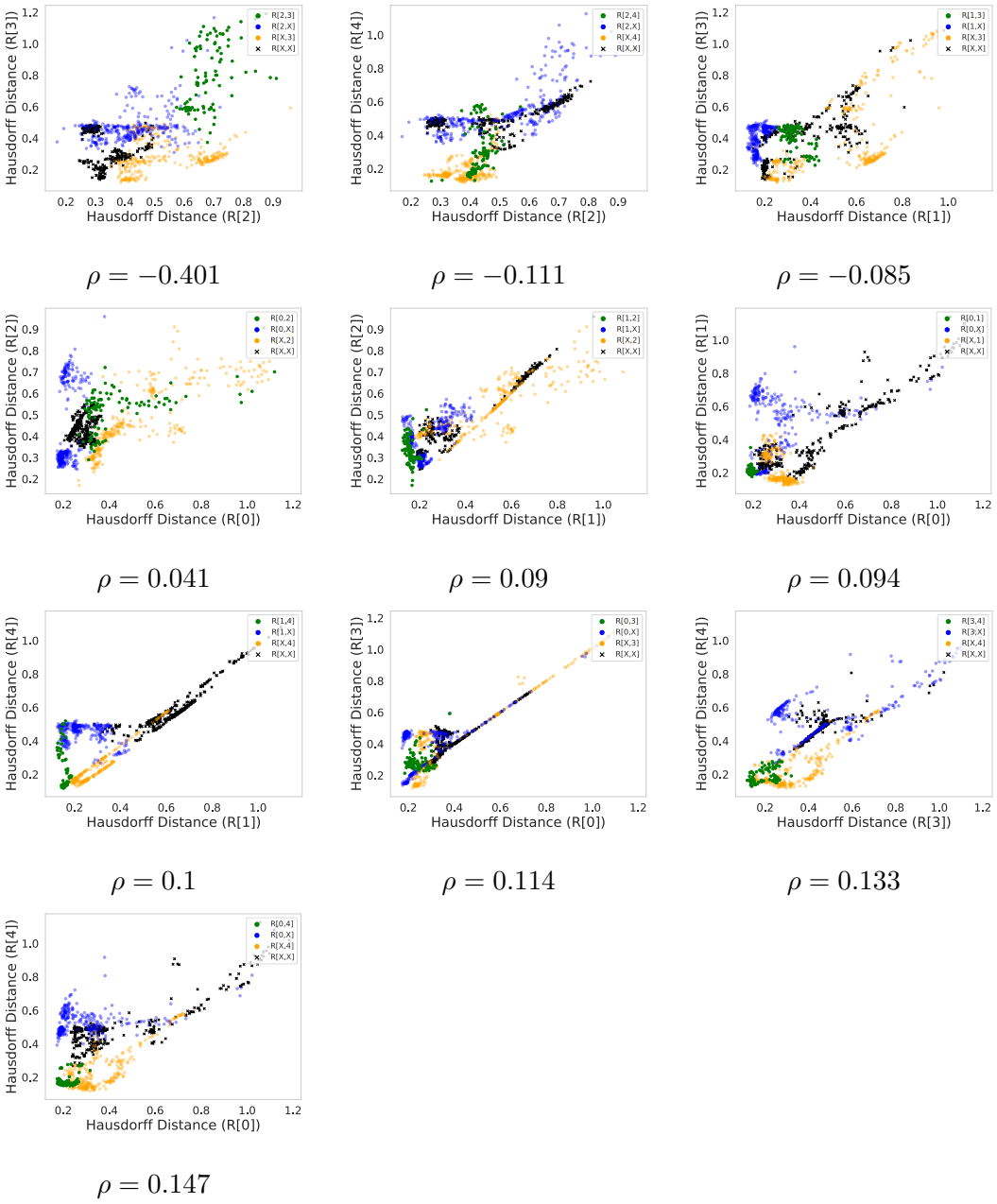
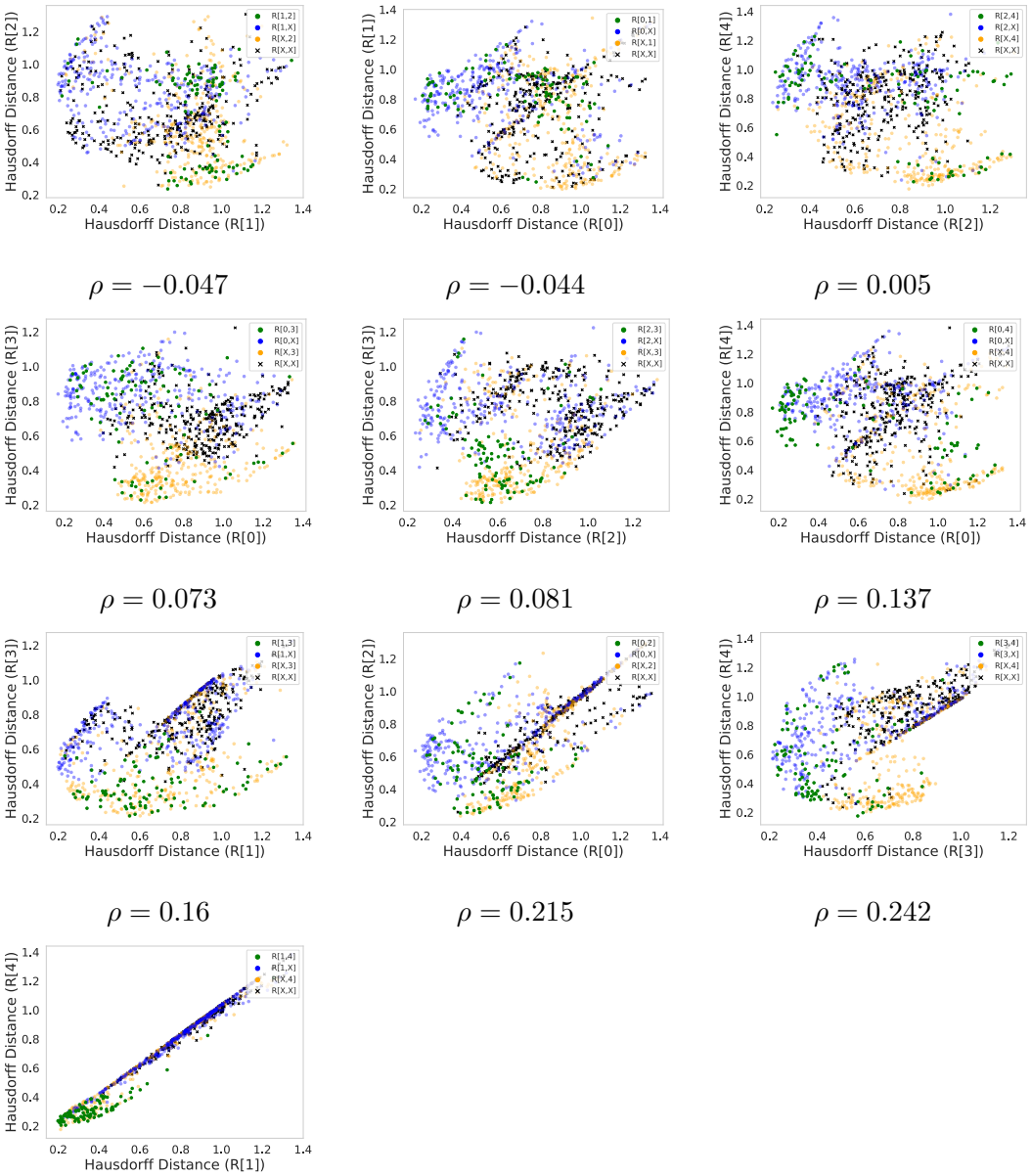


Figure A.10: Topographic maps and their associated topographic scores for each combination of features with one-hot referents

Visual - Shared Perspectives



$\rho = -0.047$ $\rho = -0.044$ $\rho = 0.005$

$\rho = 0.073$ $\rho = 0.081$ $\rho = 0.137$

$\rho = 0.16$ $\rho = 0.215$ $\rho = 0.242$

$\rho = 0.501$

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

Visual - Unshared Perspectives

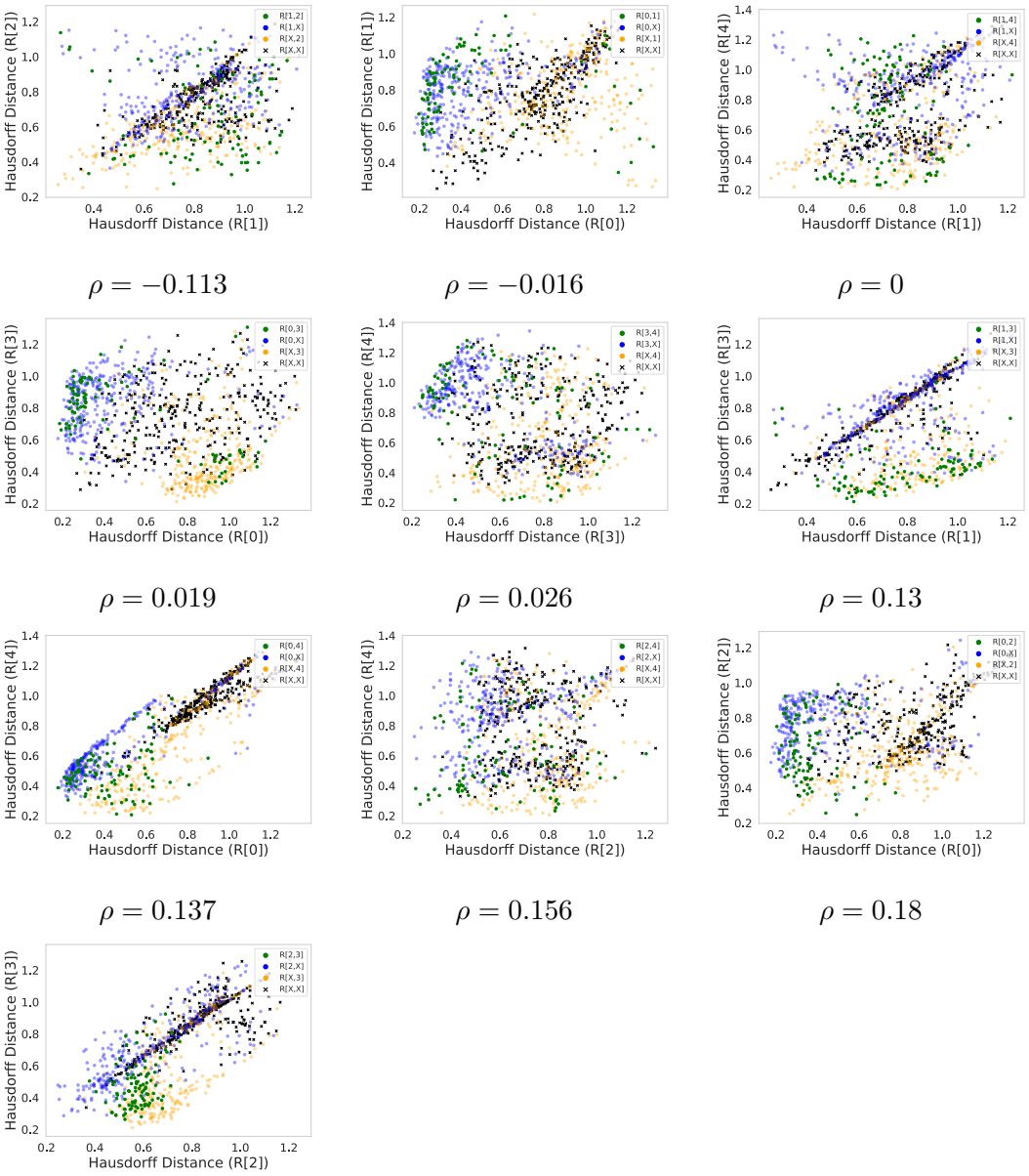


Figure A.12: Topographic maps and their associated topographic scores for each combination of features with unshared-visual referents

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

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

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

R_2 referents & descriptive utterances

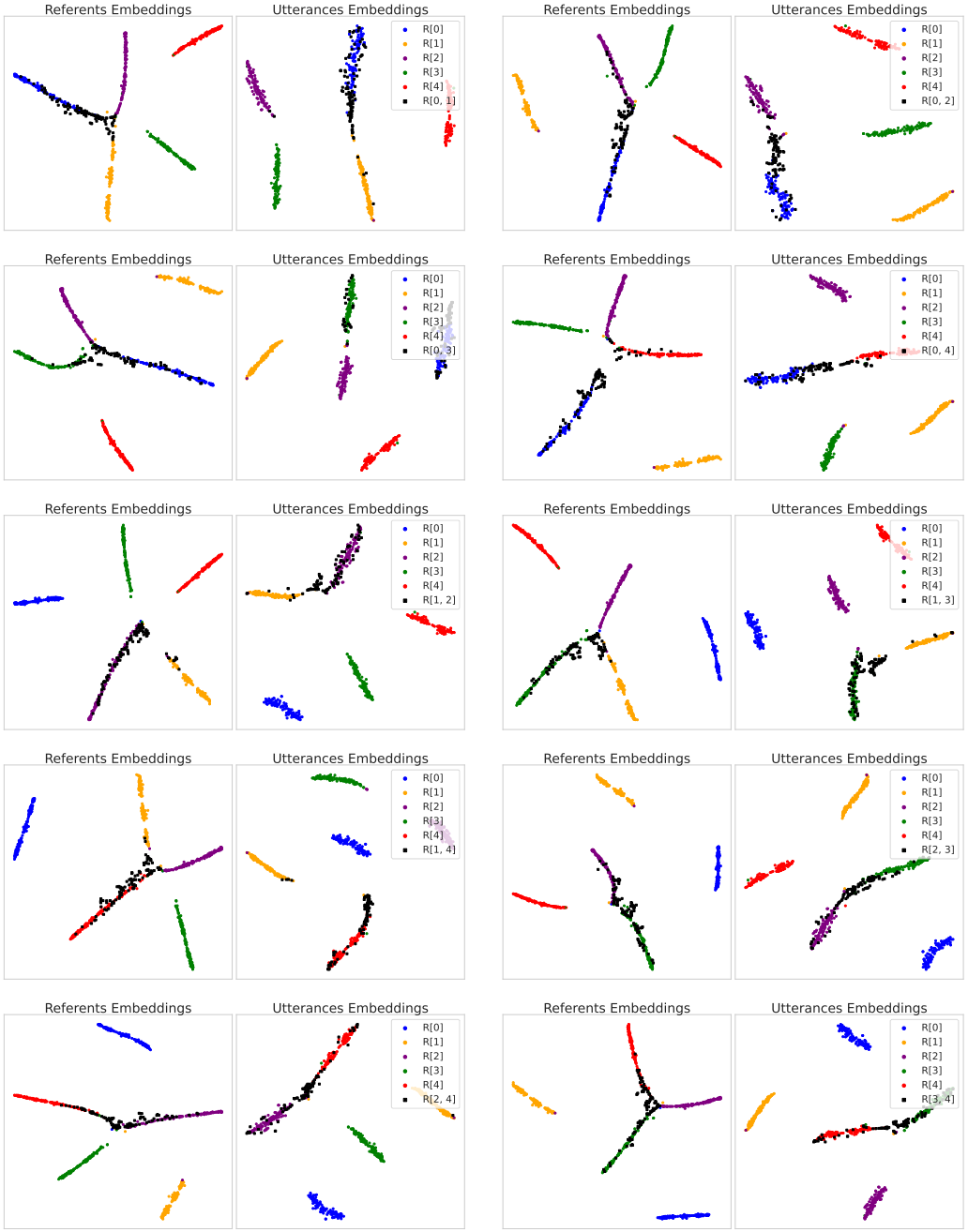


Figure A.14: **T-sne of referent and descriptive utterance embeddings.** Embeddings are computed for 100 perspectives of referents from R_2. Training conditions are unshared visual referents.

R_2 referents & discriminative utterances

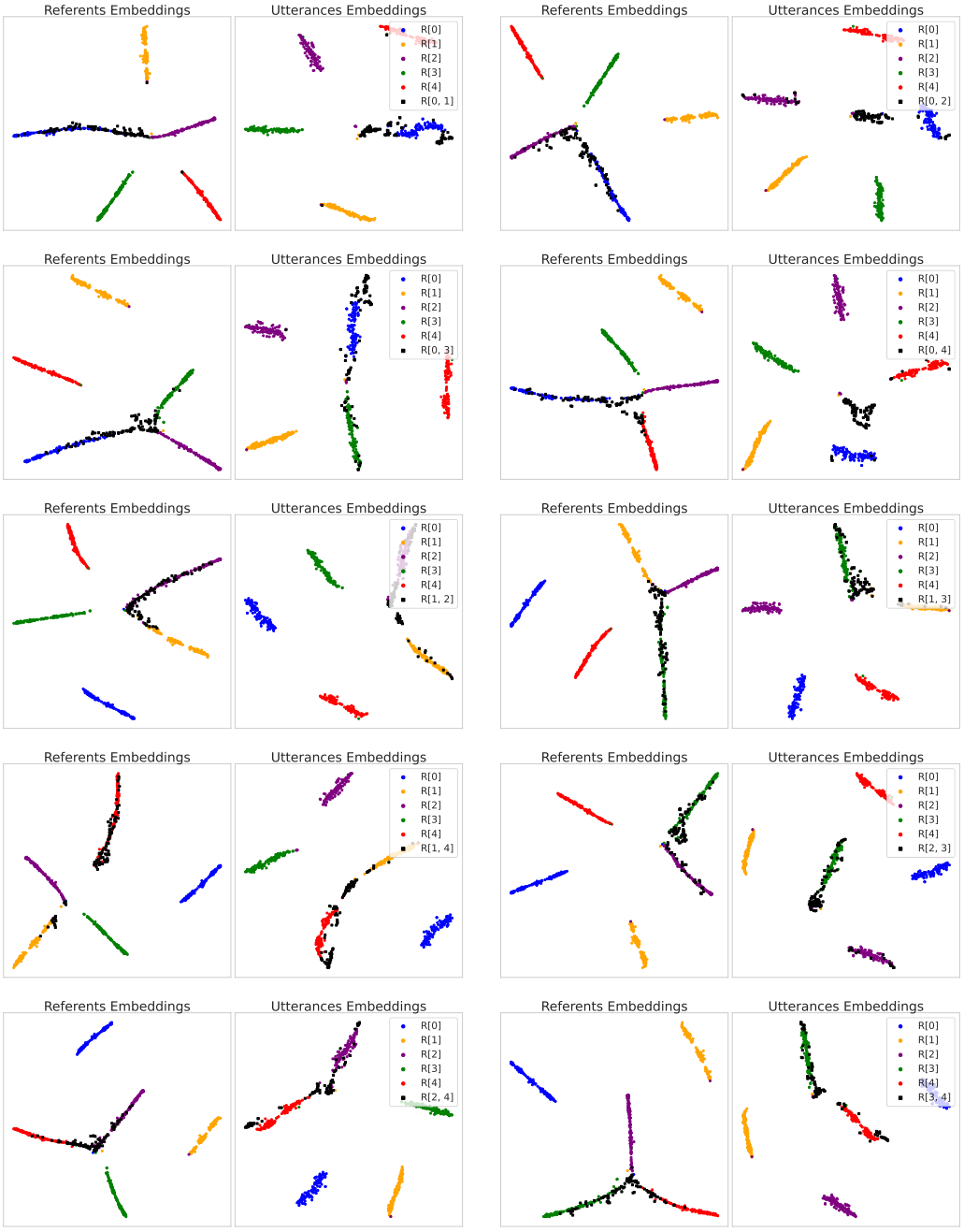


Figure A.15: **T-sne of referent and discriminative utterance embeddings.** Embeddings are computed for 100 perspectives of referents from R_2. Training conditions are unshared visual referents.

Appendix B

ABIG

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

- Section B.1 proposes derivations, implementation details and analysis related to our method.
  - Subsection B.1.1 provides additional diagrams illustrating the ABP problem and its position with respect to related settings.
  - Subsection B.1.2 proposes the full derivation of the agents' MDP.
  - Subsection **??** proposes our methods pseudo-code, algorithmic implementation details (for BC and MCTS), hyper-parameters and compute resources.
  - Subsection **??** proposes analysis that explore our method's learning mechanisms.
  - Subsection B.1.4 discusses the differences between ABP and Hierarchical/Feudal Reinforcement Learning.

B.1 Supplementary Methods

B.1.1 Supplementary Sketches

B.1.2 Analytical Description

Transition Probabilites from the architect point of view

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

$$\begin{aligned} P_{A}(s'|s,m) &= \sum_{a\in\mathcal{A}} P(s',a|s,m) \\ &= \sum_{a\in\mathcal{A}} P(s'|a,s,m)P(a|s,m) \\ &= \sum_{a\in\mathcal{A}} P_{E}(s'|a,s)\tilde{\pi}_{b}(a|s,m) \end{aligned} \tag{B.1}$$

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

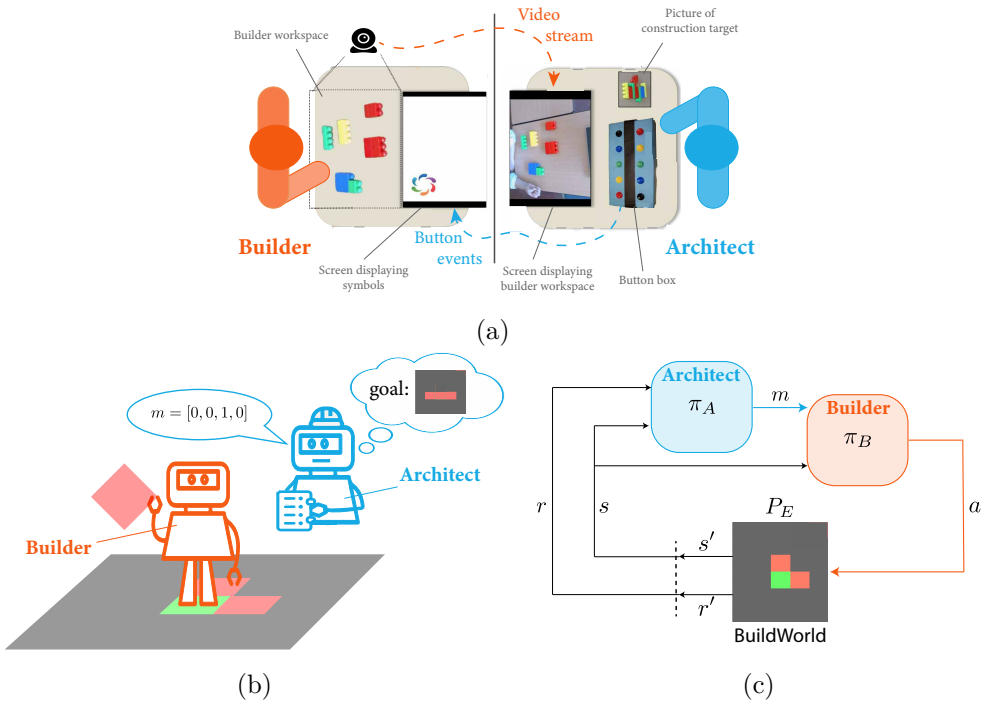
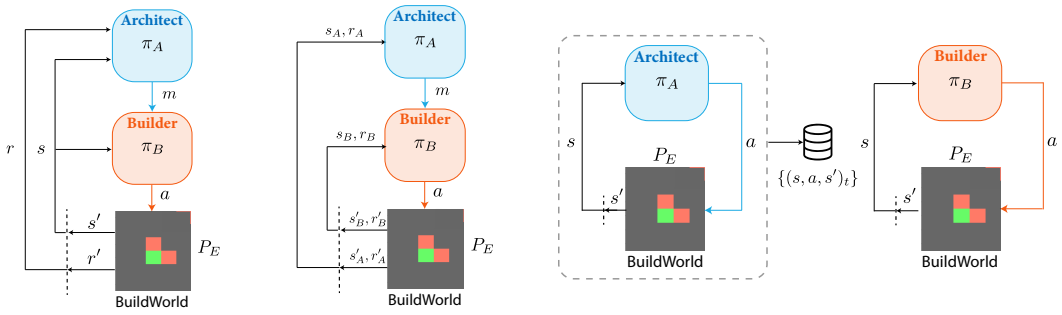


Figure B.1: (a) **Schematic view of the CoCo Game.** The architect and the builder should collaborate in order to build the construction target while located in different rooms. The architecture has a picture of the target while the builder has access to the blocks. The architect monitors the builder 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



(a) (b) (c)

Figure B.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

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

Transition function from the builder point of view

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

B.1.3 Practical Algorithm

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:

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

Monte-Carlo Tree Search

In the architect's MCTS, nodes are labeled by the 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 of 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 the parents node at depth k according to:

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

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:

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

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. (B.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 B.1: Toy experiment hyper-parameters

budget	reuse tree	max tree depth
100	true	500

Table B.2: MCTS parameters

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

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 $[5 \times 10^{-4}, 1 \times 10^{-4}, 1 \times 10^{-5}]$ for both architect and builder (vocabulary size was fixed at 6). For 'grasp' with 6 blocks the searched range is $[1 \times 10^{-3}, 5 \times 10^{-4}, 1 \times 10^{-4}]$ for the architect and $[5 \times 10^{-4}, 1 \times 10^{-4}, 5 \times 10^{-5}]$ for the

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 5.12).

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.

B.1.4 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.

Bibliography

Abbeel, P. and Ng, A. Y. Apprenticeship learning via inverse reinforcement learning. In *Proceedings of the Twenty-First International Conference on Machine Learning*, ICML '04, pp. 1, New York, NY, USA, 2004. Association for Computing Machinery. ISBN 1581138385. doi: 10.1145/1015330.1015430. URL https://doi.org/10.1145/1015330.1015430.

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.

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*, 2021.

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, 2017.

Arora, S. and Doshi, P. A survey of inverse reinforcement learning: Challenges, methods and progress. 297:103500, 2021. ISSN 0004-3702. doi: https://doi.org/10.1016/j.artint.2021.103500. URL https://www.sciencedirect.com/science/article/pii/S0004370221000515.

Ashby, W. R. Principles of the self-organizing system. In Foerster, H. V. and Jr, G. W. Z. (eds.), *Principles of Self-Organization: Transactions of the University of Illinois Symposium*, pp. 255–278. Pergamon Press, 1962.

Auer, P., Cesa-Bianchi, N., and Fischer, P. Finite-time analysis of the multiarmed bandit problem. 47(2):235–256, 2002. ISSN 1573-0565. doi: 10.1023/A:1013689704352. URL https://doi.org/10.1023/A:1013689704352.

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.

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.

Beer, R. D. A dynamical systems perspective on agent-environment interaction. *Artificial Intelligence*, 72(1):173–215, 1995. ISSN 0004-3702. doi: https://doi.org/10.1016/0004-3702(94)00005-L. URL https://www.sciencedirect.com/science/article/pii/000437029400005L.

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.

Bellman, R. Dynamic programming. *Science*, 153(3731):34–37, 1966.

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

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

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.

Besse, P., Guillouet, B., Loubes, J.-M., and François, R. Review and perspective for distance based trajectory clustering, 2015.

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.

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. Compositional syntax from cultural transmission. *Artificial Life*, 8:25–54, 2002.

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

Brown, D. S., Goo, W., and Niekum, S. Better-than-demonstrator imitation learning via automatically-ranked demonstrations. In Kaelbling, L. P., Kragic, D., and Sugiura, K. (eds.), *Proceedings of the Conference on Robot Learning*, volume 100 of *Proceedings of Machine Learning Research*, pp. 330–359. PMLR, 30 Oct–01 Nov 2020a. URL https://proceedings.mlr.press/v100/brown20a.html.

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, 2020b.

Browne, C. B., Powley, E., Whitehouse, D., Lucas, S. M., Cowling, P. I., Rohlfshagen, P., Tavener, S., Perez, D., Samothrakis, S., and Colton, S. A survey of monte carlo tree search methods. *IEEE Transactions on Computational Intelligence and AI in Games*, 4 (1):1–43, 2012. doi: 10.1109/TCIAIG.2012.2186810.

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.

Camazine, S., Deneubourg, J.-L., Franks, N. R., Sneyd, J., Theraula, G., and Bonabeau, E. *Self-Organization in Biological Systems*. Princeton University Press, Princeton, 2001. ISBN 9780691212920. doi: doi:10.1515/9780691212920. URL https://doi.org/10.1515/9780691212920.

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.

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

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.

Carruthers, P. Thinking in language?: Evolution and a modularist possibility. In *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.

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, 2019a.

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*, 2019b.

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.

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*, 2019.

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.

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. *Reflections on Language*. Number v. 10 in Pantheon Books. Pantheon Books, 1975. ISBN 9780394499567. URL https://books.google.fr/books?id=R78kAQAAMAAJ.

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.

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.

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, 2020.

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., 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, 2019.

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., Moulin-Frier, C., and Oudeyer, P.-Y. Language as a Cognitive Tool: Dall-E, Humans and Vygotskian RL Agents, March 2021. URL https://hal.archives-ouvertes.fr/hal-03159786.

Colas, C., Karch, T., Moulin-Frier, C., and Oudeyer, P.-Y. Language and Culture Internalisation for Human-Like Autotelic AI. *Nature Machine Intelligence*, 2022a.

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*, 2022b.

Crawford, V. P. and Sobel, J. Strategic information transmission. *Econometrica: Journal of the Econometric Society*, pp. 1431–1451, 1982.

Csikzentmihalyi, M. Finding flow: The psychology of engagement with everyday life. *New York: Basic*, 1997.

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.

Dayan, P. and Hinton, G. E. Feudal reinforcement learning. In *Advances in Neural Information Processing Systems 5, [NIPS Conference]*, pp. 271278, 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, 1993.

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.

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.

Dilts, R. Nlp and self-organization theory. *Anchor Point*, 9(6):14–21, 1995.

Ding, Y., Florensa, C., Abbeel, P., and Phielipp, M. Goal-conditioned imitation learning. In *Proc. of NeurIPS*, pp. 15298–15309, 2019.

Eccles, T., Bachrach, Y., Lever, G., Lazaridou, A., and Graepel, T. Biases for emergent communication in multi-agent reinforcement learning. In *NeurIPS*, volume 32, 2019.

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.

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.

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.

Ferreira, M., Conceição, H., Viriyasitavat, W., and Tonguz, O. Self-organized traffic control. pp. 85–90, 09 2010. doi: 10.1145/1860058.1860077.

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.

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.

Fujimoto, S., van Hoof, H., and Meger, D. Addressing function approximation error in actor-critic methods. In Dy, J. and Krause, A. (eds.), *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 1587–1596. PMLR, 10–15 Jul 2018. URL https://proceedings.mlr.press/v80/fujimoto18a.html.

Galantucci, B. and Garrod, S. Experimental semiotics: a review. *Frontiers in human neuroscience*, 5:11, 2011.

Glenberg, A. M. and Kaschak, M. P. Grounding language in action. *Psychonomic Bulletin & Review*, 9(3):558–565, September 2002. ISSN 1531-5320. doi: 10.3758/BF03196313. URL https://doi.org/10.3758/BF03196313.

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, 2018.

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.

Haarnoja, T., Zhou, A., Abbeel, P., and Levine, S. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In Dy, J. and Krause, A. (eds.), *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 1861–1870. PMLR, 10–15 Jul 2018. URL https://proceedings.mlr.press/v80/haarnoja18b.html.

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.

Hartikainen, K., Geng, X., Haarnoja, T., and Levine, S. Dynamical distance learning for semi-supervised and unsupervised skill discovery. In *Proc. of ICLR*, 2020.

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.

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.

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*, 2020.

Ho, J. and Ermon, S. Generative adversarial imitation learning. In Lee, D., Sugiyama, M., Luxburg, U., Guyon, I., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc., 2016. URL https://proceedings.neurips.cc/paper/2016/file/cc7e2b878868cbae992d1fb743995d8f-Paper.pdf.

Hockett, C. F. and Hockett, C. D. The origin of speech. *Scientific American*, 203(3):88–97, 1960.

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.

Hunt, J. M. Intrinsic motivation and its role in psychological development. *Nebraska symposium on motivation*, 13:189–282, 1965. URL https://cir.nii.ac.jp/crid/1571698599234799104.

Hurford, J. R. Biological evolution of the saussurean sign as a component of the language acquisition device. *Lingua*, 77:187–222, 1989.

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, J. and Lu, Z. Learning attentional communication for multi-agent cooperation. In *NeurIPS*, 2018.

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, 2019.

Kaelbling, L. P. Learning to achieve goals. In *IJCAI*, pp. 1094–1099. Citeseer, 1993.

Kaplan, F. and Oudeyer, P.-Y. In search of the neural circuits of intrinsic motivation. *Frontiers in neuroscience*, 1:17, 2007.

Karch, T., Colas, C., Teodorescu, L., Moulin-Frier, C., and Oudeyer, P.-Y. Deep sets for generalization in rl, 2020. URL https://arxiv.org/abs/2003.09443.

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.

Ke, L., Choudhury, S., Barnes, M., Sun, W., Lee, G., and Srinivasa, S. Imitation learning as f-divergence minimization, 2020.

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.

Kidd, C., Piantadosi, S. T., and Aslin, R. N. The goldilocks effect: Human infants allocate attention to visual sequences that are neither too simple nor too complex. *PLoS ONE*, 7, 2012.

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.

Kirby, S. Spontaneous evolution of linguistic structure-an iterated learning model of the emergence of regularity and irregularity. *IEEE Trans. Evol. Comput.*, 5:102–110, 2001.

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.

Klein, E., Geist, M., and Pietquin, O. Batch, Off-policy and Model-free Apprenticeship Learning. In *EWRL 2011*, pp. 1–12, Athens, Greece, September 2011. URL https://hal-supelec.archives-ouvertes.fr/hal-00660623.

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.

Kova, 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.

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.

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.

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.

Lemesle, Y., Karch, T., Laroche, R., Moulin-Frier, C., and Oudeyer, P.-Y. Emergence of shared sensory-motor graphical language from visual input, 2022. URL https://arxiv.org/abs/2210.06468.

Levy, A., Platt, R., and Saenko, K. Hierarchical reinforcement learning with hindsight. ArXiv - abs/1805.08180, 2018.

Lewis, D. K. *Convention: A Philosophical Study.* Cambridge, MA, USA: Wiley-Blackwell, 1969.

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.

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.

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.

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.

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.

Lynch, C. and Sermanet, P. Grounding language in play. ArXiv - abs/2005.07648, 2020.

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.

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.

Mcclung, J., Placì, S., Bangerter, A., Clément, F., and Bshary, R. The language of cooperation: Shared intentionality drives variation in helping as a function of group membership. *Proceedings of the Royal Society B: Biological Sciences*, 284:20171682, 09 2017. doi: 10.1098/rspb.2017.1682.

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.

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.

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 sensorymotor 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.

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.

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.

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.

Neu, G. and Szepesvári, C. Apprenticeship learning using inverse reinforcement learning and gradient methods. In *Proceedings of the Twenty-Third Conference on Uncertainty in Artificial Intelligence*, UAI'07, pp. 295302, Arlington, Virginia, USA, 2007. AUAI Press. ISBN 0974903930.

Ng, A. Y. and Russell, S. J. Algorithms for inverse reinforcement learning. In *Proceedings of the Seventeenth International Conference on Machine Learning*, ICML '00, pp. 663670, San Francisco, CA, USA, 2000. Morgan Kaufmann Publishers Inc. ISBN 1558607072.

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.

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.

Oliphant, M. and Batali, J. Learning and the emergence of coordinated communication. *Center for Research on Language Newsletter*, 11, 03 1997.

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.

OroojlooyJadid, A. and Hajinezhad, D. A review of cooperative multi-agent deep reinforcement learning, 2019. URL https://arxiv.org/abs/1908.03963.

Osa, T., Pajarinen, J., Neumann, G., Bagnell, J. A., Abbeel, P., Peters, J., et al. An algorithmic perspective on imitation learning. *Foundations and Trends® in Robotics*, 7 (1-2):1–179, 2018.

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 Smith, L. B. How evolution may work through curiosity-driven developmental process. *Topics in Cognitive Science*, 8(2):492–502, 2016.

Pashevich, A., Schmid, C., and Sun, C. Episodic Transformer for Vision-and-Language Navigation. *ArXiv – abs/2105.06453*, 2021.

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.

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.

Pfeifer, R., Lungarella, M., and Iida, F. Self-organization, embodiment, and biologically inspired robotics. *Science*, 318(5853):1088–1093, 2007. doi: 10.1126/science.1145803. URL https://www.science.org/doi/abs/10.1126/science.1145803.

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.

Polyak, B. T. and Juditsky, A. B. Acceleration of stochastic approximation by averaging. *SIAM Journal on Control and Optimization*, 30(4):838–855, 1992. doi: 10.1137/0330046.

Pomerleau, D. A. Alvinn: An autonomous land vehicle in a neural network. In Touretzky, D. (ed.), *Advances in Neural Information Processing Systems*, volume 1. Morgan-Kaufmann, 1988. URL https://proceedings.neurips.cc/paper/1988/file/812b4ba287f5ee0bc9d43bbf5bbe87fb-Paper.pdf.

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, 2000.

Puterman, M. L. *Markov decision processes: discrete stochastic dynamic programming*. John Wiley & Sons, 2014.

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.

Ratliff, N. D., Bagnell, J. A., and Zinkevich, M. A. Maximum margin planning. In *Proceedings of the 23rd International Conference on Machine Learning*, ICML '06, pp. 729736, New York, NY, USA, 2006. Association for Computing Machinery. ISBN 1595933832. doi: 10.1145/1143844.1143936. URL https://doi.org/10.1145/1143844.1143936.

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.

Rohlfing, K. J., Wrede, B., Vollmer, A.-L., and Oudeyer, P.-Y. An Alternative to Mapping a Word onto a Concept in Language Acquisition: Pragmatic Frames. *Frontiers in Psychology*, 7, 2016. ISSN 1664-1078.

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.

Ross, S., Gordon, G., and Bagnell, D. A reduction of imitation learning and structured prediction to no-regret online learning. In Gordon, G., Dunson, D., and Dudík, M. (eds.), *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, volume 15 of *Proceedings of Machine Learning Research*, pp. 627–635, Fort Lauderdale, FL, USA, 11–13 Apr 2011. PMLR. URL https://proceedings.mlr.press/v15/ross11a.html.

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.

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.

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. Learning from demonstration. In Mozer, M., Jordan, M., and Petsche, T. (eds.), *Advances in Neural Information Processing Systems*, volume 9. MIT Press, 1996. URL https://proceedings.neurips.cc/paper/1996/file/68d13cf26c4b4f4f932e3eff990093ba-Paper.pdf.

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, 2015.

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, 1991.

Schulman, J., Levine, S., Abbeel, P., Jordan, M., and Moritz, P. Trust region policy optimization. In Bach, F. and Blei, D. (eds.), *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pp. 1889–1897, Lille, France, 07–09 Jul 2015. PMLR. URL https://proceedings.mlr.press/v37/schulman15.html.

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and Klimov, O. Proximal policy optimization algorithms, 2017. URL https://arxiv.org/abs/1707.06347.

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.

Sharma, A., Gu, S., Levine, S., Kumar, V., and Hausman, K. Dynamics-aware unsupervised discovery of skills. In *Proc. of ICLR*, 2020.

Shridhar, M., Thomason, J., Gordon, D., Bisk, Y., Han, W., Mottaghi, R., Zettlemoyer, L., and Fox, D. ALFRED: A benchmark for interpreting grounded instructions for everyday tasks. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020, Seattle, WA, USA, June 13-19, 2020*. IEEE, 2020.

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.

Steels, L. A self-organizing spatial vocabulary. *Artificial life*, 2(3):319–332, 1995a.

Steels, L. The synthetic modeling of language origins. *Evolution of Communication Journal*, 1, 10 1997. doi: 10.1075/eoc.1.1.02ste.

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.

Sukhbaatar, S., Szlam, A. D., and Fergus, R. Learning multiagent communication with backpropagation. In *NIPS*, 2016.

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. *Reinforcement learning: An introduction*. MIT press, 2018.

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.

Tomasello, M. *The Cultural Origins of Human Cognition*. Harvard University Press, 1999. ISBN 978-0-674-00582-2.

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., Carpenter, M., Call, J., Behne, T., and Moll, H. Understanding and Sharing Intentions: The Origins of Cultural Cognition. *Behavioral and brain sciences*, 2005.

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.

von Foerster, H. *On Self-Organizing Systems and Their Environments*, pp. 1–19. Springer New York, New York, NY, 2003. ISBN 978-0-387-21722-2. doi: 10.1007/0-387-21722-3_1. URL https://doi.org/10.1007/0-387-21722-3_1.

Vygotsky, L. S. Play and Its Role in the Mental Development of the Child. *Soviet Psychology*, 1933.

Vygotsky, L. S. *Thought and Language*. MIT press, 1934.

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.

Watkins, C. J. C. H. and Dayan, P. Q-learning. 8(3):279–292, 1992. ISSN 1573-0565. doi: 10.1007/BF00992698. URL https://doi.org/10.1007/BF00992698.

Wellman, H. M. *The child's theory of mind*. The MIT Press, 1992.

Whorf, B. L. *Language, Thought, and Reality: Selected Writings of Benjamin Lee Whorf*. MIT press, 1956.

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.

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.

Zhang, Y., Abbeel, P., and Pinto, L. Automatic curriculum learning through value disagreement. In *Proc. of NeurIPS*, 2020.

Zhu, C., Dastani, M., and Wang, S. A survey of multi-agent reinforcement learning with communication, 2022.

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.

Ziebart, B., Maas, A., Bagnell, J., and Dey, A. Maximum entropy inverse reinforcement learning. In *AAAI Conference on Artificial Intelligence*, 2008.

Zuidema, W. and De Boer, B. The evolution of combinatorial phonology. *Journal of Phonetics*, 37(2):125–144, 2009.

Zwaan, R. A. and Madden, C. J. *Embodied Sentence Comprehension*, pp. 224245. Cambridge University Press, 2005. doi: 10.1017/CBO9780511499968.010.