Hierarchically Organized Latent Modules for Exploratory Search in Morphogenetic Systems

Mayalen Etcheverry, Clément Moulin-Frier, Pierre-Yves Oudeyer

Hierarchically Organized Latent Modules for Exploratory Search in Morphogenetic Systems

Mayalen Etcheverry[*], **Clément Moulin-Frier**, **Pierre-Yves Oudeyer**
Flowers Team
Inria, Univ. Bordeaux, Ensta ParisTech (France)
{mayalen.etcheverry,clement.moulin-frier,pierre-yves.oudeyer}@inria.fr

Abstract

Self-organization of complex morphological patterns from local interactions is a fascinating phenomenon in many natural and artificial systems. In the artificial world, typical examples of such morphogenetic systems are cellular automata. Yet, their mechanisms are often very hard to grasp and so far scientific discoveries of novel patterns have primarily been relying on manual tuning and ad hoc exploratory search. The problem of automated *diversity-driven* discovery in these systems was recently introduced [26, 62], highlighting that two key ingredients are autonomous exploration and unsupervised representation learning to describe "relevant" degrees of variations in the patterns. In this paper, we motivate the need for what we call *Meta-diversity* search, arguing that there is not a unique ground truth *interesting* diversity as it strongly depends on the final observer and its motives. Using a continuous game-of-life system for experiments, we provide empirical evidences that relying on monolithic architectures for the behavioral embedding design tends to bias the final discoveries (both for hand-defined and unsupervisedly-learned features) which are unlikely to be aligned with the interest of a final end-user. To address these issues, we introduce a novel *dynamic* and *modular* architecture that enables unsupervised learning of a hierarchy of diverse representations. Combined with intrinsically motivated goal exploration algorithms, we show that this system forms a discovery assistant that can efficiently adapt its diversity search towards preferences of a user using only a very small amount of user feedback.

1 Introduction

Self-organisation refers to a broad range of pattern-formation processes in which globally-coherent structures spontaneously emerge out of local interactions. In many natural and artificial systems, understanding this phenomenon poses fundamental challenges [1]. In biology, *morphogenesis*, where cellular populations self-organize into a structured morphology, is a prime example of complex developmental process where much remains to be assessed. Since Turing's influential paper *"The Chemical Basis of Morphogenesis"* [74], many mathematical and computational models of morphogenetic systems have been proposed and extensively studied [24]. For example, cellular automata abstract models like Conway's Game of Life (GoL), despite their apparent simplicity, generate a wide range of life-like structures ranging from Turing-like patterns reminiscent of animal skins stripes and spots [1] to localized autonomous patterns showing key behaviors like self-replication [38].

A modern goal of morphogenesis research has become the manipulation, exploration and control of self-organizing systems. With the recent advances in synthetic biology and high-precision roboticized experimental conditions, potential applications range from automated material design [72], drug discovery [66] and regenerative medicine [18] to unraveling the chemical origins of life [26, 48]. Yet,

[*]Source code, videos and additional results can be found at `http://mayalenE.github.io/holmes/`

34th Conference on Neural Information Processing Systems (NeurIPS 2020), Vancouver, Canada.

even for simple artificial systems like Conway's Game of Life, we do not fully grasp the mapping from local rules to global structures nor have a clear way to represent and classify the discovered structures. While scientific discoveries of novel patterns have primarily been relying on manual tuning and ad hoc exploratory search, the growing potential of data-driven search coupled with machine learning (ML) algorithms allows us to rethink our approach. In fact, contemporary work opened the path toward three promising directions in applying ML to morphogenesis research, namely in 1) the design of richer computational models, coupling neural networks data-structures [23] with deep-learning techniques to *learn* the update rules of the system, allowing to directly achieve key properties such as self-regeneration and self-replication [49]; 2) by formulating morphological search as a reinforcement learning problem where an artificial agent learns to control the morphologies self-assembly to achieve a certain task such as locomotion [54]; and 3) formulating the problem of automated *diversity-driven* discovery in morphogenetic systems, and proposing to transpose intrinsically-motivated exploration algorithms, coming from the field of developmental robotics, to this new kind of problem [26, 62].

Pure objective-driven search is unlikely to scale to complex morphogenetic systems. It can even be inapplicable when scientists do not know how to characterize desired behaviours from raw observations (reward definition problem), or merely aim to find novel patterns. As an illustration of these challenges, it took 40 years before the first replicator "spaceship" pattern was spotted in Conway's Game of Life. We believe that *diversity-driven* approaches can be powerful discovery tools [26, 62], and can potentially be coupled with objective-driven searches [12, 53, 58]. Recent families of machine learning have shown to be effective at creating *behavioral diversity*, namely Novelty Search (NS) [41, 42] and Quality-Diversity (QD) [16, 58] coming from the field of evolutionary robotics; and intrinsically-motivated goal-directed exploration processes (IMGEP) [2, 21] coming from developmental robotics. A known critical part of these algorithms, is that they require the definition of a *behavioral characterization* (BC) feature space which formalizes the "interesting" degrees of behavioral variation in the environment [57]. So far, this behavior space has either been hand-defined in robotic scenarios [16, 21] or unsupervisedly learned with deep auto-encoders directly from raw observations [15, 39, 51, 55, 56]. While deep auto-encoders have shown to recover the "ground truth" factor of variations in simple generative datasets [5], it is impossible (and not desirable) to recover *all* the degrees of variations in the targeted self-organizing systems.

In this paper, we follow the proposed experimental testbed of Reinke et al. (2020) [62] on a continuous game-of-life system (Lenia, [6]). We provide empirical evidence that the discoveries of an IMGEP operating in a *monolithic* BC space are highly-diverse in that space, yet tend to be poorly-diverse in other potentially-interesting BC spaces. This draws several limitations when it comes to applying such system as a tool for assisting discovery in morphogenetic system, as the suggested discoveries are unlikely to align with the interests of a end-user. How to build an artificial "discovery assistant" learning to generate diverse patterns in the eyes of a future, yet unknown, observer? A central challenge in that direction is to extend the standard notion of *diversity*, where an agent discovers diverse patterns in a monolithic BC space, to what we call *meta-diversity*, where an agent incrementally learns diverse behavioral characterization spaces and discovers diverse patterns within each of them. A second key challenge is to build exploration strategies that can quickly adapt to the preferences of a human end-user, while requiring minimal feedback. To address these challenges, we propose a novel model architecture for unsupervised representation learning with Hierarchically Organized Latent Modules for Exploratory Search, called *HOLMES*. We compare the behavioral characterizations learned by an IMGEP equipped with HOLMES hierarchy of goal space representations to an IMGEP using a single monolithic goal space representation, and the resulting discoveries. We consider two end-user models respectively interested in two types of diversities (diverse spatially localized and diverse turing-like patterns), and show that a monolithic IMGEP will make discoveries that are strongly uneven in relation to these user preferences, while IMGEP-HOLMES is better suited to escape this bias by learning divergent feature representations. Additionally, we show how HOLMES can be efficiently *guided* to drive the search toward those two types of *interesting* diversities with very little amount of (simulated) user feedback.

Our contributions are threefold. We introduce the novel objective of *meta-diversity* search in the context of automated discovery in morphogenetic systems. We propose a *dynamic* and *modular* model architecture for unsupervised learning of *diverse* representations, which, to our knowledge, is the first work that proposes to progressively grow the capacity of the agent visual world model into an organized hierarchical representation. We show how this architecture can easily be leveraged to *drive* exploration, opening interesting perspectives for the integration of a human in the loop.

2 Problem Formulation and Motivation for *Meta-Diversity* Search

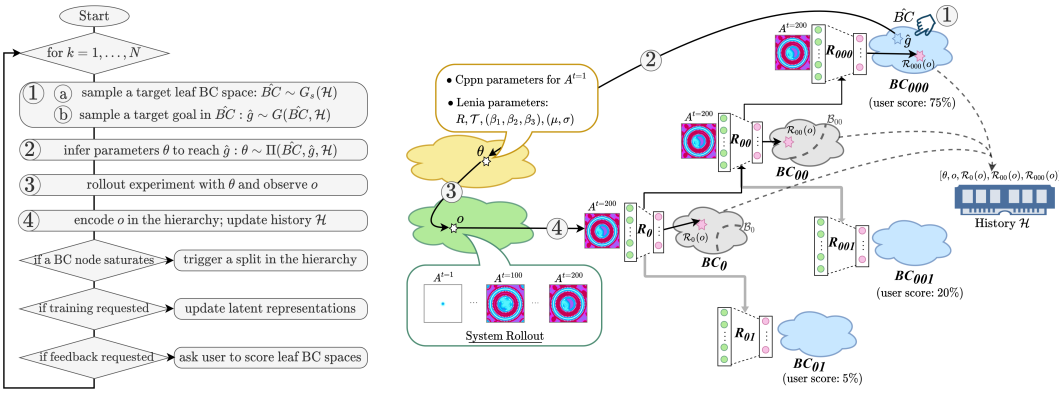
We summarize the problem of automated discovery of morphogenetic systems as formulated in [62], on a continuous game of life environment example. We identify limits of this formulation and associated approach. The novel process of *meta-diversity* search is proposed within this framework.

A morphogenetic system: Lenia Morphogenetic systems are characterized by an *initial state* ($A^{t=1}$, seed of the system) as well as a set of *local update rules* that are iteratively applied to evolve the state of the system through time ($A^{t=1} \rightarrow \ldots \rightarrow A^{t=T}$). Typically observed from raw images, the emerging patterns depend on a set of *controllable parameters* θ that, for each experimental run, condition the system initial state and update rules. We use Lenia cellular automaton [6, 7], a continuous generalization to Conway's Game of Life, as testbed. It can generate a wide range of complex behaviors and life-like structures, as testified by the collection of "species" that were manually identified and classified in [6]. To organize the experimental study of exploration algorithms, several restrictions were proposed [62]: the lattice resolution is fixed to 256×256 and the evolution is stopped after 200 time steps. Moreover, only the system final state is observed ($A^{t=200}$), focusing the search on morphological appearance traits and leaving out dynamical traits (yet see section 4 for side-effect discoveries of interesting dynamics). The controllable parameters include a 7-dimensional set of parameters controlling Lenia's update rule as well as parameters governing the generation of the initial state $A^{t=1}$. Compositional-pattern producing networks (CPPN) [71] are used to produce the initial patterns.

Automated discovery problem The standard automated discovery problem ([62]) consists of generating a maximally-diverse set of observations through sequential experimentation of controllable parameters θ. Each controllable parameter vector θ generates a rollout $A^{t=1} \rightarrow \ldots \rightarrow A^{t=T}$ of the morphogenetic system leading to an observation $o(\theta) = A^{t=T}$. This observation is encoded as a vector $r = R(o)$ in a BC space representing interesting features of the observation (e.g. based on the color or frequency content of the observation image). Given a budget of N experiments, the objective of the automated discovery problem is to sample a set of controllable parameters Θ where $\{R(o(\theta))|\theta \in \Theta\}$ maximally covers the BC space.

Problem definition: Meta-Diversity Search The standard automated discovery problem defined above assumes that the intuitive notion of diversity can be captured within a single BC space (what we call a monolithic representation). However, as our results will show, maximizing the diversity in one BC space may lead to poor diversity in other, possibly-interesting, BC spaces. Thus, using a single representation space to drive exploration algorithms limits the scope of their discoveries, as well as the scope of their external evaluation. To address this limit, we formulate the novel process of *meta-diversity* search: in an outer loop, one aims to learn a *diverse* set of behavioral characterizations (called the *meta-diversity*); then in an inner loop, one aims to discover a set of maximally diverse patterns in each of the BC spaces (corresponding to the standard notion of *diversity* in previous work). The objective of this process is to enable continuous seeking of novel niches of diversities while being able to quickly adapt the search toward a new unknown *interesting* diversity. Here, a successful discovery assistant agent is one which can leverage its diverse BCs to specialize efficiently towards a particular type of diversity, corresponding to the initially unknown preferences of an end-user, and expressed through simple and very sparse feedback.

Proposed approach: IMGEP Agent with modular BC spaces A goal-directed intrinsically-motivated exploration process (IMGEP) was used for the parameter sampling strategy in [62]. After an initialization phase, the IMGEP strategy iterates through 1) sampling a goal in a learned BC space R, conditioned on the explicit memory of the system $\mathcal{H}$ and based on the goal-sampling strategy $g \sim G(H)$; 2) sampling a set of parameters θ for the next rollout to achieve that goal, based on its parameter-sampling policy $\Pi = Pr(\theta; g, H)$; 3) let the system rollout, observe outcome o, and store the resulting $(\theta, o, R(o))$ triplet in an explicit memory $\mathcal{H}$. However, the behavioral characterization was based on a monolithic representation. Although being learned, this representation was limited to capture diversity in a single BC space and was therefore unable to perform meta-diversity search as defined above. To solve this problem, we introduce a modular architecture where a hierarchy of behavioral characterization spaces is progressively constructed, allowing flexible representations and intuitive guidance during the discovery process.



(a) IMGEP exploratory loop

(b) HOLMES hierarchy of diverse representations

Figure 1: IMGEP-HOLMES framework integrates a goal-based intrinsically-motivated exploration process (IMGEP) with the incremental learning of a hierarchy of behavioral characterization spaces (HOLMES). HOLMES unsupervisedly clusters and encodes discovered patterns into the different nodes of the hierarchy of representations. The architecture of HOLMES and clustering part is detailed in section 3.1. The exploratory loop and its interaction with the hierarchy of behavioral characterization (BC) spaces, enabling *meta-diversity search*, is detailed in section 3.2.

3 Hierarchically Organized Latent Modules for Exploratory Search

3.1 HOLMES: Architecture

HOLMES is a dynamic architecture that "starts small" both on the task data distribution and on the network memory capacity, following the intuition of Elman (1993) [19]. A hierarchy of embedding networks $\mathcal{R} = \{\mathcal{R}_i\}$ is actively expanded to accommodate to the different niches of patterns discovered in the environment, resulting in a progressively deeper hierarchy of specialized BC spaces. The architecture has 4 main components: (i) a base *module* embedding neural network, (ii) a *saturation* signal that triggers the instantiating of new nodes in the hierarchy, (iii) a *boundary* criteria that unsupervisedly clusters the incoming patterns into the different modules; and (iv) a *connection-scheme* that allows feature-wise transfer from a parent module to its children.
We refer to Figure 1 and Figure 6 (section A, appendix) for an illustration of the following.

In this paper, we use a variational auto-encoder (VAE) [33] as base *module*. The hierarchy starts with a single VAE $\mathcal{R}_0$ that is incrementally trained on the incoming data, encoding it into its latent characterization space BC_0. A split is triggered when a node of the hierarchy *saturates* i.e. when the reconstruction loss of its VAE reaches a plateau, with additional conditions to prevent premature splitting (minimal node population and a minimal number of training steps). Each time a node gets *saturated*, the split procedure is executed as follows. First, the parameters of the parent $\mathcal{R}_p$ are frozen and two child networks $\mathcal{R}_{p0}$ and $\mathcal{R}_{p1}$ are instantiated with their own neural capacity. Besides, additional learnable layers called *lateral connections* are instantiated between the parent and child VAE feature-maps, drawing inspiration from *Progressive Neural Networks* (PNN) [64]. Here, these layers allow the child VAE to reuse its parent knowledge while learning to characterize novel dissimilar features in its BC space (see section 4.2 and appendix D.3 for an ablation study). Finally, a boundary $\mathcal{B}_p$ is fitted in the parent frozen embedding space BC_p and kept fixed for the rest of the exploration. In this paper, the boundary is unsupervisedly fitted with K-means by assigning two clusters from the points that are currently in the node's BC space. Based on this boundary criteria, incoming observations forwarding through $\mathcal{R}_p$ will be either redirected through the left child $\mathcal{R}_{p0}$ or thought the right child $\mathcal{R}_{p1}$. After the split, training continues: leaf node VAEs as well as their incoming lateral connections are incrementally trained on their own niches of patterns while non-leaf nodes are frozen and serve to redirect the incoming patterns.

In HOLMES, the clustering of the different patterns is central to learn *diverse* BC spaces. The clustering is influenced by the choice of the module and connections training strategy, that determines the latent distribution of patterns in the latent space, and by the clustering algorithm itself. Note that the genericity of HOLMES architecture (agnostic to the base module) allows many other design choices to be considered in future work, and that current choices are discussed in Appendix A.1.

3.2 IMGEP-HOLMES: Interaction with the Exploration Process

The goal space of an IMGEP is usually defined as the BC space of interest, with a representation based on a monolithic architecture $\mathcal{R}$ [39, 55]. In this paper, we propose a variant where the IMGEP operates in a hierarchy of goal spaces $\{BC_i\}$, where observations and hence goals are encoded at different levels or granularity, as defined by HOLMES embedding hierarchy $\{\mathcal{R}_i\}$. The exploration process iterates N times through steps 1-to-4, as illustrated in Figure 1. In this section we detail the implementation choices for each step. We refer to algorithm 1 in appendix for a general pseudo-code.

1) The goal-sampling strategy is divided in two sub-steps. **a)** Sample a target BC space $\hat{BC}_i$ with a goal space sampling distribution G_s. Here, the agent is given an additional degree of control allowing to prioritize exploration in certain nodes of the hierarchy (and therefore on a subset population of patterns). In this paper we considered two setups for the goal space sampling strategy: (i) a *non-guided* variant where the target BC space is sampled uniformly over all the leaf nodes and (ii) a *guided* variant where, after each split in the hierarchy we "pause" exploration and ask for evaluator feedback to assign an *interest score* to the different leaf modules. The guided variant simulates a human observer that could, by visually browsing at few representative images per module and simply "click" and score the leaf nodes with the preferred discoveries. Then during exploration, the agent selects over the leaf goal spaces with softmax sampling on the assigned score probabilities. **b)** Sample a target goal $\hat{g}$ in the selected BC space with a goal sampling distribution G. In this paper, we use a uniform sampling strategy: the goal is uniformly sampled in the hypercube envelope of currently-reached goals. Because the volume of the hypercube is larger than the cloud of currently-reached goals, uniform sampling in the hypercube incentivizes to search in unexplored area outside this cloud (this is equivalent to novelty search in the selected BC space). Note that other goal-sampling mechanisms can be considered within the IMGEP framework [2].

2) The parameter-sampling strategy Π generates the CPPN-generated initial state and Lenia's update rule in 2 steps: (i) given the goal $\hat{g} \in \hat{BC}_i$, select parameters $\hat{\theta}$ in $\mathcal{H}$ whose corresponding outcome $R_i(o)$ is closest to $\hat{g}$ in $\hat{BC}_i$; (ii) mutate the selected parameters by a random process $\theta = \text{MUTATION}(\hat{\theta})$ (see appendix C.2 for implementation details).

3) Rollout experiment with parameters θ and observe outcome o.

4) Forward o top-down through the hierarchy and retrieve respective embeddings $\{R_i(o)\}$ along the path. Append respective triplets $\{(\theta, o, R_i(o))\}$ to the episodic memory $\mathcal{H}$.

HOLMES online training. The data distribution collected by the IMGEP agent directly influences HOLMES splitting procedure and training procedure by determining which nodes get populated and when. In this paper, we incrementally train the goal space hierarchy every $N_T = 100$ exploration step for $E = 100$ epochs on the observations collected in the history $\mathcal{H}$. Importance sampling is used at each training stage, giving more weight to recently discovered patterns. We refer to appendix C.3 for implementation details on HOLMES training procedure.

4 Experimental Results

In this section we compare the results of an IMGEP equipped with a goal space based on different types of BCs. We denote **IMGEP-X** an IMGEP operating in a goal space X (X can be e.g. an analytical BC space based on Fourier descriptors, or a modular representation learned by the HOLMES architecture as in section 3.2). In order to evaluate meta-diversity, we make the distinction between the BC used as the goal space of an IMGEP and the BC used for evaluating the level of diversity reached by that IMGEP. For example, we might want to evaluate the diversity reached by **IMGEP-HOLMES** in a BC space based on Fourier descriptors. For quantitative evaluation of the diversity in a given BC, the BC space is discretized with $n = 20$ bins per dimension, and the diversity is measured as the number of bins in which at least one explored entity falls (details are provided in Appendix B.1.2). Each experiment below consists of $N = 5000$ runs starting with $N_{init} = 1000$ initial random explorations runs. For all algorithms, we conduct 10 repetitions of the exploration experiment with different random seeds. Please refer to appendix (sections B and C) for all details on the evaluation procedure and experimental settings. Additionally, the source code and full database of discoveries are provided on the project website.

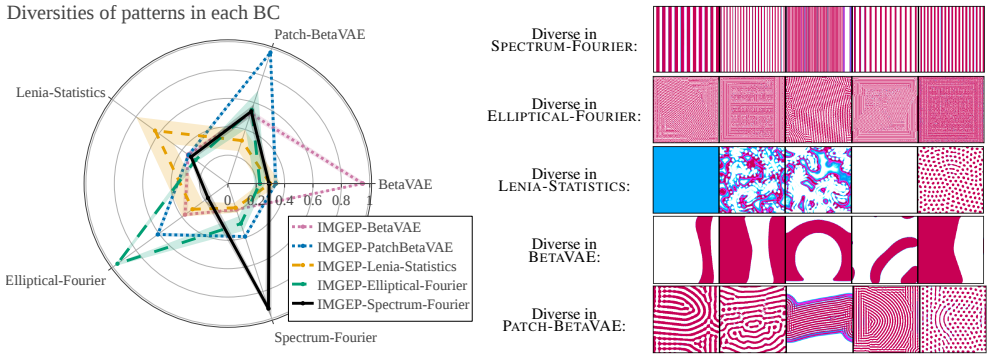


Figure 2: Although IMGEPs succeed to reach a high-diversity in their respective BC space, they are poorly-diverse in all the others. (left) Diversity for all IMGEP variants measured in each analytic BC space. For better visualisation the resulting diversities are divided by the maximum along each axis. Mean and std-deviation shaded area curves are depicted. (right). Examples of patterns discovered by the IMGEPs that are consider diverse in their respective BC space. See Appendix B.1.1 for details.

4.1 Does maximizing diversity in a given BC space lead to high diversity in other BC spaces?

We construct 5 BC spaces with 8 dimensions each, among which 3 representations are predefined and 2 are pre-learned on a database of Lenia patterns. The exploration discoveries of an IMGEP equipped with the different BCs as (fixed) goal space are evaluated and compared in figure 2. The two first BCs rely on Fourier descriptors, intended to characterize the frequencies of textures (Spectrum-Fourier) and shape of closed contours (Elliptical-Fourier [36]), typically used in cellular-automata [47] and biology [14, 80]. The third BC relies on statistical features that were proposed in the original Lenia paper [6] to describe the activation of patterns. The fourth uses a set of features unsupervisedly learned by a β-VAE [5] on a large database of Lenia patterns, as proposed in [62]. Because β-VAE can poorly represent high-frequency patterns, another variant trained on cropped patches is proposed.

Limits of monolithic goal spaces The results of Figure 2 suggest that, if we could have a theoretical BC model that aligns with what a user considers as diverse under the form of a goal space, the IMGEP should be efficient in finding a diversity of patterns in that space. In practice however, constructing such a BC is very challenging, if not infeasible. Each BC was carefully designed or unsupervisedly learned to represent what could be "relevant" factors of variations and yet, the IMGEP seems to exploit degrees of variations that might not be aligned with what we had in mind when constructing such BCs. Spectrum-Fourier is a clear example that was constructed to describe textures (in a general sense) but where the discoveries exhibit only vertical-stripe patterns with all kind of frequencies.

4.2 What is the impact of modularity in incremental learning of goal space(s)?

Baselines We compare **IMGEP-VAE** which uses a monolithic VAE as goal space representation to **IMGEP-HOLMES** which is defined in section 3. HOLMES expansion is stopped after 11 splits (resulting in a hierarchy of 23 VAEs) and uses small-capacity modules and connections, such that its final capacity is still smaller than the monolithic VAE. Other variants for the monolithic architecture and training strategies are considered in Appendix D.2.

Learning to characterize different niches We use representational similarity analysis (RSA) [35] to quantify how much the representations embeddings (encoded behaviors) evolve through the exploration inner loop (Figure 3). Different metrics have been proposed to compute the representational similarity, here we use the linear centered kernel alignment (CKA) index [34] (see Appendix B.2). Results show that the learned features of the monolithic VAE stop evolving after only 2 training stages, i.e. 200 explored patterns. This suggests that, even though the VAE is *incrementally trained* during exploration (at the difference of the pretrained variants in section 4.1), it still fails to adapt to the different niches of patterns which will lead to limited discoveries. However, RSA results suggest that HOLMES succeeds to learn features that are highly dissimilar from one module to another, which allows to target discovery of a *meta-diversity*. An ablation study (see section D.3 of appendix) highlighted the importance of HOLMES *lateral connections* to escape the VAE learning biases.

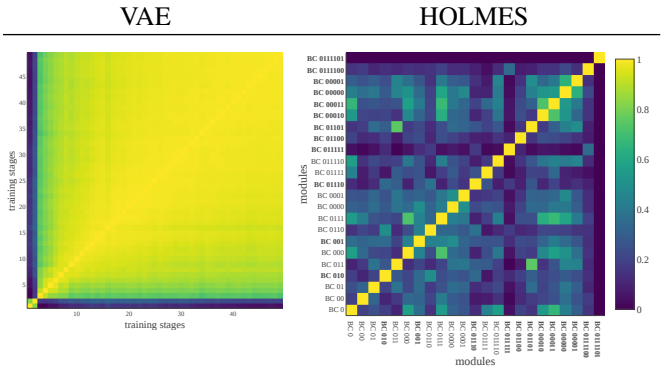


Figure 3: RSA similarity index between 0 (dark blue, not similar at all) and 1 (yellow, identical). (VAE) representations are compared in time between the different training stages. (HOLMES) representations are compared at the *end* of exploration between the different modules. Leafs are depicted in bold and modules are ordered by their creation time (left-to-right in x-axis). See Appendix D.1 for full temporal analysis and statistical results.

Learning to explore different niches Qualitative browsing through the discoveries confirms that IMGEP-HOLMES is able to explore diverse niches, allowing the discovery of very interesting behaviors in Lenia from quite unexpected types of patterns. To further investigate these discoveries, the exploration was prolonged for 10000 additional runs (without expanding more the hierarchy). Not only many of the "lifeform" patterns of the *species* identified in [6] were discovered, but it allowed to assist to the *birth* of these creatures from fluid-like structures that have shown capable of *pattern-emission* (behavior which was, to our knowledge, never discovered in 2D Lenia). Example of our discoveries can be seen in Figure 4 and on the website `http://mayalenE.github.io/holmes/`.

4.3 Can we drive the search toward an *interesting* diversity?

Two categories of patterns have been extensively studied in cellular-automata, known as Spatially-Localized Patterns (SLP) and Turing-Like Patterns (TLP) (Figure 4). We investigate if our *discovery assistant* search can be guided to specialize toward a diversity of either SLPs or TLPs. For experimentation, we propose to use a *simulated* user, preferring either SLPs or TLPs, and a *proxy* evaluation of diversity tailored to these preferences. For *guidance*, the classifiers of "animals" (SLP) and "non-animals" (TLP) from [62] are used to score the different nodes in IMGEP-HOLMES with the number of SLP (or TLP) that fall in that node at split time (see section 3.2). This simulates preferences of a user toward either SLPs or TLPs, who would use a GUI to score the patterns found in each leaves of IMGEP-HOLMES. The total number of user interventions is 11 (one per split) with, for each intervention, an average of 6 "clicks" (scores), which represents very sparse feedback. For evaluation, an experiment with a human evaluator has been conducted for selecting the BC (among the 5 proposed in section 4.1) that correlates the most with the evaluator judgement of what represents a diversity of SLP and a diversity of TLP. $\text{BC}_{\text{ELLIPTICAL-FOURIER}}$ was designed as the best proxy space for evaluating the diversity of SLP (98% agreement score) and $\text{BC}_{\text{LENIA-STATISTICS}}$ was designated for TLP (92% agreement). Experiment details are provided in appendix B.3.

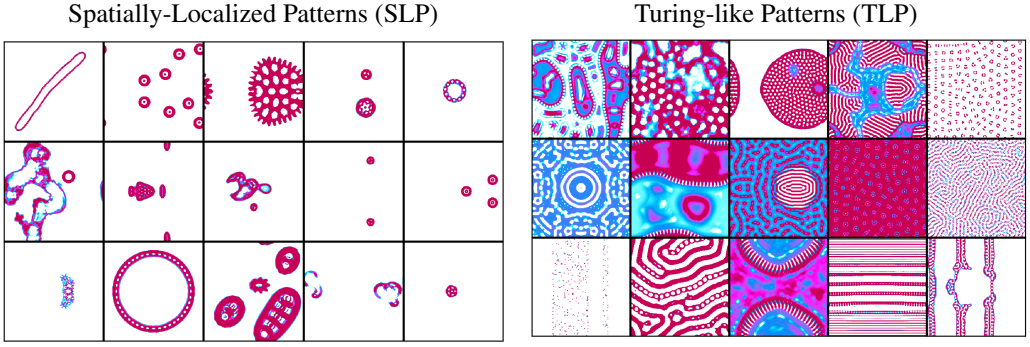


Figure 4: (Left) SLP are autonomous stable patterns, that show interesting behaviors such as locomotion and metamorphosis(shape-shifting). (Right) TLP patterns are characterized by an unlimited spatial growth resembling reaction-diffusion pattern-formation of fronts, spirals, stripes and dissipative solitons. The displayed patterns where *autonomously* discovered in Lenia [6] by IMGEP-HOLMES (without guidance) and, considered by us (human evaluator), as *interesting*.

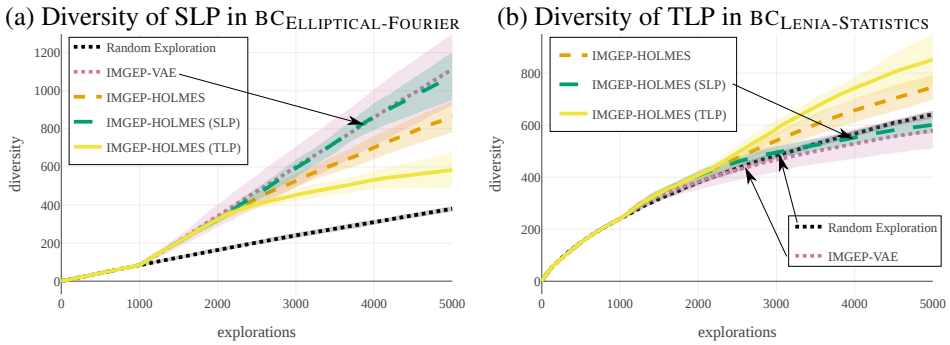


Figure 5: Depicted is the diversity discovered by the algorithms during exploration. The discovered patterns classified as SLP in (a) (resp TLP in (b)) are projected in $\text{BC}_{\text{ELLIPTICAL-FOURIER}}$ space in (a) (resp $\text{BC}_{\text{LENIA-STATISTICS}}$ in (b)), where the binning-based measure is used. Mean and std-deviation shaded area curves are depicted.

Baselines Non-guided IMGEP variants (section 4.2) are compared with guided IMGEP variants. The results are compared to Random Exploration baseline (where parameters θ are randomly sampled for the 5000 explorations) which serves as reference for the *default* diversity found in Lenia (and by all algorithms during the first 1000 explorations).

Results The results in Figure 5 show that the bias of the monolithic VAE allows IMGEP-VAE to find a high diversity of SLPs but leads to poor diversity of TLPs. When non-guided, IMGEP-HOLMES finds a higher diversity than Random Exploration both for SLPs and TLPs. When guided, IMGEP-HOLMES can even further increase the diversity in the category of interest.

5 Related Work

Machine Learning for Automated Discovery in Science Many work successfully applied ML for discovery in chemistry [17, 25, 60], physics [4, 46] and biology [27, 32]. However, they focus the search on predefined structures with target properties and not, to our knowledge, on diversity search.

Diversity Search Diversity-driven approaches in ML are presented in the introduction. In the QD literature, the work of Pugh et al. (2016) [59] relates to what we frame as *meta-diversity* search. In a maze environment where 2 sets of hand-defined BC are provided (one for agent position and one for agent direction), they show that driving the search with the two BC sets simultaneously leads to higher probability to discover *quality* behavior (meaning here solving the task) than with a single *unaligned* BC. However, BCs are predefined and fixed, limiting the generalisation to complex environments.

State Representation Learning Many approaches have been proposed for state representation learning in reinforcement learning [43]. As presented in the introduction, goal-directed approaches generally rely on deep generative models such as VAEs [51, 55]. Others tune the representation to achieve target properties at the feature level such as disentanglement [29] or linear predictability [76]. This includes coupling with predictive forward and inverse models [28, 29, 31, 53, 75, 79] or priors of independent controllability [39, 73]. However, they all rely on a single embedding space, where all the observed instances are mapped to the same set of features.

Continual Unsupervised Representation Learning Recent work also proposed to dynamically expand the network capacity of a VAE [40, 45, 61]. Similarities and differences with HOLMES are discussed in Appendix E. However, these approaches were applied to passively observed datasets, either targeting unsupervised clustering of sequentially-received class of images or disentanglement of factors of variations in generative datasets.

Interactive Exploration of Patterns Interactive evolutionary computation (IEC) [37, 68, 69] aims to integrate external human feedback to explore high complexity pattern spaces, via intuitive interfaces. However, the user must provide feedback at each generation by individually selecting interesting patterns for the next generation; whereas our framework requires much sparser feedback.

6 Discussion

As stated in the Introduction, our contributions in this paper are threefold. First, in section 2, we introduced the novel objective of *meta-diversity* search in the context of automated discovery in morphogenetic systems. Then, in section 3, we proposed a *dynamic* and *modular* model architecture for meta-diversity search through unsupervised learning of *diverse* representations. Finally, in section 4, we showed that search can easily be guided toward complex pattern preferences of a simulated end user, using very little user feedback.

To our knowledge, HOLMES is the first algorithm that proposes to progressively grow the capacity of the agent visual world model into an organized hierarchical representation. There are however several limitations to be addressed in future works. The architecture remains quite *rigid* in the way it is isolating the different niches of patterns (binary tree with frozen boundaries) whereas other approaches, further leveraging human feedback, could be envisaged.

The question whether *machines* can really help scientists for crucial discoveries in Science, although appealing, is still an open question [3]. We believe that machine learning algorithms integrating flexible modular representation learning with intrinsically-motivated goal exploration processes for *meta-diversity* search are very promising directions. As an example, despite the limitations mentioned above, IMGEP-HOLMES was able to discover many types of solitons including unseen pattern-emitting lifeforms in less than 15000 training steps without guidance, when their existence remained an open question raised in the original Lenia paper [6].

Broader Impact Statement

We introduced methods that can be used as tools to help human scientists discover novel structures in complex dynamical systems. While experiments presented in this article were performed using an artificial system (continuous cellular automaton), they also target to be used for automated discovery of novel structures in fields ranging from biology to physics. As an example, Grizou et al. [26] recently showed how IMGEPs can be used to automate chemistry experiments addressing fundamental questions related to the origins of life (how oil droplets may self-organize into proto-cellular structures), leading to new insights about oil droplet chemistry. As experiments in Grizou et al. used a single pre-defined BC, one can expect that the new approach presented in this paper may boost the efficiency of its use in bio-physical systems, that could include systems related to design of new materials or new drugs. As a tool enabling scientist to better understand the space of dynamics of such systems, we believe it could help them better understand how to leverage such dynamics for societally useful purposes, and avoid negative effects, e.g. due to unpredicted self-organized dynamics.

However, technological and scientific discoveries might have a considerable impact in modern societies. Introducing machine decisions in the process should therefore be done with great responsibility, taking care of carefully identifying and balancing the biases inherent to any ML algorithms. The methods proposed in this paper constitute a first step in this direction by quantitatively measuring the influence of biases, in both predefined and learned BC spaces, on the algorithm discoveries. With an increasing interest in ML for automated discovery, it will be fundamental to to improve and extend these methods in the near future.

Besides, by releasing our code, we believe that we help efforts in reproducible science and allow the wider community to build upon and extend our work in the future.

Acknowledgements and Disclosure of Funding

We would like to thank Chris Reinke and Bert Chan for useful inputs and discussions for the paper, as well as Jonathan Grizou for valuable suggestions.

The authors acknowledge support from the Human Frontiers Science Program (Collaborative Research Grant RGP0018/2016) and from Inria.

References

[1] Philip Ball. *The self-made tapestry: pattern formation in nature*. Oxford University Press Oxford, 1999.

[2] Adrien Baranes and Pierre-Yves Oudeyer. Active learning of inverse models with intrinsically motivated goal exploration in robots. *Robotics and Autonomous Systems*, 61(1):49–73, 2013.

[3] Marta Bertolaso and Fabio Sterpetti. *A Critical Reflection on Automated Science*. Springer, 2020.

[4] JS Bloom, JW Richards, PE Nugent, RM Quimby, MM Kasliwal, DL Starr, D Poznanski, EO Ofek, SB Cenko, NR Butler, et al. Automating discovery and classification of transients and variable stars in the synoptic survey era. *Publications of the Astronomical Society of the Pacific*, 124(921):1175, 2012.

[5] Christopher P Burgess, Irina Higgins, Arka Pal, Loic Matthey, Nick Watters, Guillaume Desjardins, and Alexander Lerchner. Understanding disentangling in β-vae. *arXiv preprint arXiv:1804.03599*, 2018.

[6] Bert Wang-Chak Chan. Lenia-biology of artificial life. *Complex Systems*, 28(3):251–286, 2019.

[7] Bert Wang-Chak Chan. Lenia and expanded universe. *Artificial Life Conference Proceedings*, (32):221–229, 2020. doi: 10.1162/isal_a_00297. URL `https://www.mitpressjournals.org/doi/abs/10.1162/isal_a_00297`.

[8] Gal Chechik, Varun Sharma, Uri Shalit, and Samy Bengio. Large scale online learning of image similarity through ranking. *Journal of Machine Learning Research*, 11(Mar):1109–1135, 2010.

[9] Ricky T. Q. Chen, Xuechen Li, Roger Grosse, and David Duvenaud. Isolating sources of disentanglement in variational autoencoders. In *Advances in Neural Information Processing Systems*, 2018.

[10] Simon YY Chen, Pete E Lestrel, W John S Kerr, and John H McColl. Describing shape changes in the human mandible using elliptical fourier functions. *The European Journal of Orthodontics*, 22(3):205–216, 2000.

[11] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. *arXiv preprint arXiv:2002.05709*, 2020.

[12] Cédric Colas, Olivier Sigaud, and Pierre-Yves Oudeyer. GEP-PG: Decoupling Exploration and Exploitation in Deep Reinforcement Learning Algorithms. In *International Conference on Machine Learning (ICML)*, Stockholm, Sweden, July 2018. URL `https://hal.inria.fr/hal-01890151`.

[13] James S Cope, Paolo Remagnino, Sarah Barman, and Paul Wilkin. Plant texture classification using gabor co-occurrences. In *International Symposium on Visual Computing*, pages 669–677. Springer, 2010.

[14] James S Cope, David Corney, Jonathan Y Clark, Paolo Remagnino, and Paul Wilkin. Plant species identification using digital morphometrics: A review. *Expert Systems with Applications*, 39(8):7562–7573, 2012.

[15] Antoine Cully. Autonomous skill discovery with quality-diversity and unsupervised descriptors. In *Proceedings of the Genetic and Evolutionary Computation Conference*, pages 81–89, 2019.

[16] Antoine Cully, Jeff Clune, Danesh Tarapore, and Jean-Baptiste Mouret. Robots that can adapt like animals. *Nature*, 521(7553):503–507, 2015.

[17] Vasilios Duros, Jonathan Grizou, Weimin Xuan, Zied Hosni, De-Liang Long, Haralampos N Miras, and Leroy Cronin. Human versus robots in the discovery and crystallization of gigantic polyoxometalates. *Angewandte Chemie*, 129(36):10955–10960, 2017.

[18] Kevin Dzobo, Nicholas Ekow Thomford, Dimakatso Alice Senthebane, Hendrina Shipanga, Arielle Rowe, Collet Dandara, Michael Pillay, and Keolebogile Shirley Caroline M Motaung. Advances in regenerative medicine and tissue engineering: Innovation and transformation of medicine. *Stem cells international*, 2018, 2018.

[19] Jeffrey L Elman. Learning and development in neural networks: The importance of starting small. *Cognition*, 48(1):71–99, 1993.

[20] Jan Flusser. Moment invariants in image analysis. In *proceedings of world academy of science, engineering and technology*, volume 11, pages 196–201. Citeseer, 2006.

[21] Sébastien Forestier, Rémy Portelas, Yoan Mollard, and Pierre-Yves Oudeyer. Intrinsically motivated goal exploration processes with automatic curriculum learning. *arXiv preprint arXiv:1708.02190*, 2017.

[22] Martin Friess and Michel Baylac. Exploring artificial cranial deformation using elliptic fourier analysis of procrustes aligned outlines. *American Journal of Physical Anthropology: The Official Publication of the American Association of Physical Anthropologists*, 122(1):11–22, 2003.

[23] William Gilpin. Cellular automata as convolutional neural networks. *Physical Review E*, 100 (3):032402, 2019.

[24] Chad M Glen, Melissa L Kemp, and Eberhard O Voit. Agent-based modeling of morphogenetic systems: Advantages and challenges. *PLoS computational biology*, 15(3), 2019.

[25] Rafael Gómez-Bombarelli, Jennifer N Wei, David Duvenaud, José Miguel Hernández-Lobato, Benjamín Sánchez-Lengeling, Dennis Sheberla, Jorge Aguilera-Iparraguirre, Timothy D Hirzel, Ryan P Adams, and Alán Aspuru-Guzik. Automatic chemical design using a data-driven continuous representation of molecules. *ACS central science*, 4(2):268–276, 2018.

[26] Jonathan Grizou, Laurie J Points, Abhishek Sharma, and Leroy Cronin. A curious formulation robot enables the discovery of a novel protocell behavior. *Science advances*, 6(5):eaay4237, 2020.

[27] Sean R Hackett, Edward A Baltz, Marc Coram, Bernd J Wranik, Griffin Kim, Adam Baker, Minjie Fan, David G Hendrickson, Marc Berndl, and R Scott McIsaac. Learning causal networks using inducible transcription factors and transcriptome-wide time series. *Molecular systems biology*, 16(3):e9174, 2020.

[28] Danijar Hafner, Timothy Lillicrap, Ian Fischer, Ruben Villegas, David Ha, Honglak Lee, and James Davidson. Learning latent dynamics for planning from pixels. *arXiv preprint arXiv:1811.04551*, 2018.

[29] Irina Higgins, Arka Pal, Andrei Rusu, Loic Matthey, Christopher Burgess, Alexander Pritzel, Matthew Botvinick, Charles Blundell, and Alexander Lerchner. Darla: Improving zero-shot transfer in reinforcement learning. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 1480–1490. JMLR. org, 2017.

[30] Ming-Kuei Hu. Visual pattern recognition by moment invariants. *IRE transactions on information theory*, 8(2):179–187, 1962.

[31] Maximilian Karl, Maximilian Soelch, Justin Bayer, and Patrick Van der Smagt. Deep variational bayes filters: Unsupervised learning of state space models from raw data. *arXiv preprint arXiv:1605.06432*, 2016.

[32] Ross D King, Kenneth E Whelan, Ffion M Jones, Philip GK Reiser, Christopher H Bryant, Stephen H Muggleton, Douglas B Kell, and Stephen G Oliver. Functional genomic hypothesis generation and experimentation by a robot scientist. *Nature*, 427(6971):247–252, 2004.

[33] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv:1312.6114*, 2013.

[34] Simon Kornblith, Mohammad Norouzi, Honglak Lee, and Geoffrey Hinton. Similarity of neural network representations revisited. *arXiv preprint arXiv:1905.00414*, 2019.

[35] Nikolaus Kriegeskorte, Marieke Mur, and Peter A Bandettini. Representational similarity analysis-connecting the branches of systems neuroscience. *Frontiers in systems neuroscience*, 2:4, 2008.

[36] Frank P Kuhl and Charles R Giardina. Elliptic fourier features of a closed contour. *Computer graphics and image processing*, 18(3):236–258, 1982.

[37] WB Langdon. Pfeiffer–a distributed open-ended evolutionary system. In *AISB*, volume 5, pages 7–13. Citeseer, 2005.

[38] Christopher G Langton. Self-reproduction in cellular automata. *Physica D: Nonlinear Phenomena*, 10(1-2):135–144, 1984.

[39] Adrien Laversanne-Finot, Alexandre Pere, and Pierre-Yves Oudeyer. Curiosity driven exploration of learned disentangled goal spaces. volume 87 of *Proceedings of Machine Learning Research*, pages 487–504. PMLR, 29–31 Oct 2018. URL `http://proceedings.mlr.press/v87/laversanne-finot18a.html`.

[40] Soochan Lee, Junsoo Ha, Dongsu Zhang, and Gunhee Kim. A neural dirichlet process mixture model for task-free continual learning. *arXiv preprint arXiv:2001.00689*, 2020.

[41] Joel Lehman and Kenneth O Stanley. Exploiting open-endedness to solve problems through the search for novelty. In *ALIFE*, pages 329–336, 2008.

[42] Joel Lehman and Kenneth O Stanley. Abandoning objectives: Evolution through the search for novelty alone. *Evolutionary computation*, 19(2):189–223, 2011.

[43] Timothée Lesort, Natalia Díaz-Rodríguez, Jean-Franois Goudou, and David Filliat. State representation learning for control: An overview. *Neural Networks*, 108:379–392, 2018.

[44] Pete E Lestrel. *Fourier descriptors and their applications in biology*. Cambridge University Press, 1997.

[45] Zhiyuan Li, Jaideep Vitthal Murkute, Prashnna Kumar Gyawali, and Linwei Wang. Progressive learning and disentanglement of hierarchical representations. *arXiv preprint arXiv:2002.10549*, 2020.

[46] Shuaihua Lu, Qionghua Zhou, Yixin Ouyang, Yilv Guo, Qiang Li, and Jinlan Wang. Accelerated discovery of stable lead-free hybrid organic-inorganic perovskites via machine learning. *Nature communications*, 9(1):1–8, 2018.

[47] Jeaneth Machicao, Lucas C Ribas, Leonardo FS Scabini, and Odermir M Bruno. Cellular automata rule characterization and classification using texture descriptors. *Physica A: Statistical Mechanics and its Applications*, 497:109–117, 2018.

[48] Jan Meisner, Xiaolei Zhu, and Todd J Martínez. Computational discovery of the origins of life, 2019.

[49] Alexander Mordvintsev, Ettore Randazzo, Eyvind Niklasson, and Michael Levin. Growing neural cellular automata. *Distill*, 2020. doi: 10.23915/distill.00023. https://distill.pub/2020/growing-ca.

[50] WH Müller. Fourier transforms and their application to the formation of textures and changes of morphology in solids. In *IUTAM Symposium on Transformation Problems in Composite and Active Materials*, pages 61–72. Springer, 1998.

[51] Ashvin V Nair, Vitchyr Pong, Murtaza Dalal, Shikhar Bahl, Steven Lin, and Sergey Levine. Visual reinforcement learning with imagined goals. In *Advances in Neural Information Processing Systems*, pages 9191–9200, 2018.

[52] João Camargo Neto, George E Meyer, David D Jones, and Ashok K Samal. Plant species identification using elliptic fourier leaf shape analysis. *Computers and electronics in agriculture*, 50(2):121–134, 2006.

[53] Deepak Pathak, Pulkit Agrawal, Alexei A Efros, and Trevor Darrell. Curiosity-driven exploration by self-supervised prediction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 16–17, 2017.

[54] Deepak Pathak, Christopher Lu, Trevor Darrell, Phillip Isola, and Alexei A Efros. Learning to control self-assembling morphologies: a study of generalization via modularity. In *Advances in Neural Information Processing Systems*, pages 2292–2302, 2019.

[55] Alexandre Péré, Sébastien Forestier, Olivier Sigaud, and Pierre-Yves Oudeyer. Unsupervised Learning of Goal Spaces for Intrinsically Motivated Goal Exploration. In *ICLR2018 - 6th International Conference on Learning Representations*, Vancouver, Canada, April 2018. URL `https://hal.archives-ouvertes.fr/hal-01891758`.

[56] Vitchyr H Pong, Murtaza Dalal, Steven Lin, Ashvin Nair, Shikhar Bahl, and Sergey Levine. Skew-fit: State-covering self-supervised reinforcement learning. *arXiv preprint arXiv:1903.03698*, 2019.

[57] Justin K Pugh, Lisa B Soros, Paul A Szerlip, and Kenneth O Stanley. Confronting the challenge of quality diversity. In *Proceedings of the 2015 Annual Conference on Genetic and Evolutionary Computation*, pages 967–974, 2015.

[58] Justin K Pugh, Lisa B Soros, and Kenneth O Stanley. Quality diversity: A new frontier for evolutionary computation. *Frontiers in Robotics and AI*, 3:40, 2016.

[59] Justin K Pugh, Lisa B Soros, and Kenneth O Stanley. Searching for quality diversity when diversity is unaligned with quality. In *International Conference on Parallel Problem Solving from Nature*, pages 880–889. Springer, 2016.

[60] Paul Raccuglia, Katherine C Elbert, Philip DF Adler, Casey Falk, Malia B Wenny, Aurelio Mollo, Matthias Zeller, Sorelle A Friedler, Joshua Schrier, and Alexander J Norquist. Machine-learning-assisted materials discovery using failed experiments. *Nature*, 533(7601):73–76, 2016.

[61] Dushyant Rao, Francesco Visin, Andrei Rusu, Razvan Pascanu, Yee Whye Teh, and Raia Hadsell. Continual unsupervised representation learning. In *Advances in Neural Information Processing Systems*, pages 7645–7655, 2019.

[62] Chris Reinke, Mayalen Etcheverry, and Pierre-Yves Oudeyer. Intrinsically motivated discovery of diverse patterns in self-organizing systems. In *International Conference on Learning Representations*, 2020.

[63] John C Russ. Image processing. In *Computer-assisted microscopy*, pages 33–69. Springer, 1990.

[64] Andrei A Rusu, Neil C Rabinowitz, Guillaume Desjardins, Hubert Soyer, James Kirkpatrick, Koray Kavukcuoglu, Razvan Pascanu, and Raia Hadsell. Progressive neural networks. *arXiv preprint arXiv:1606.04671*, 2016.

[65] Samuel M Scheiner, Evsey Kosman, Steven J Presley, and Michael R Willig. Decomposing functional diversity. *Methods in Ecology and Evolution*, 8(7):809–820, 2017.

[66] Petra Schneider, W Patrick Walters, Alleyn T Plowright, Norman Sieroka, Jennifer Listgarten, Robert A Goodnow, Jasmin Fisher, Johanna M Jansen, José S Duca, Thomas S Rush, et al. Rethinking drug design in the artificial intelligence era. *Nature Reviews Drug Discovery*, pages 1–12, 2019.

[67] Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 815–823, 2015.

[68] Jimmy Secretan, Nicholas Beato, David B D'Ambrosio, Adelein Rodriguez, Adam Campbell, Jeremiah T Folsom-Kovarik, and Kenneth O Stanley. Picbreeder: A case study in collaborative evolutionary exploration of design space. *Evolutionary computation*, 19(3):373–403, 2011.

[69] Joel Simon. Ganbreeder, code: `https://github.com/joel-simon/ganbreeder`. 2018.

[70] Fethi Smach, Cedric Lemaître, Jean-Paul Gauthier, Johel Miteran, and Mohamed Atri. Generalized fourier descriptors with applications to objects recognition in svm context. *Journal of mathematical imaging and Vision*, 30(1):43–71, 2008.

[71] Kenneth O Stanley. Exploiting regularity without development. In *AAAI Fall Symposium: Developmental Systems*, page 49, 2006.

[72] Daniel P Tabor, Loïc M Roch, Semion K Saikin, Christoph Kreisbeck, Dennis Sheberla, Joseph H Montoya, Shyam Dwaraknath, Muratahan Aykol, Carlos Ortiz, Hermann Tribukait, et al. Accelerating the discovery of materials for clean energy in the era of smart automation. *Nature Reviews Materials*, 3(5):5–20, 2018.

[73] Valentin Thomas, Jules Pondard, Emmanuel Bengio, Marc Sarfati, Philippe Beaudoin, Marie-Jean Meurs, Joelle Pineau, Doina Precup, and Yoshua Bengio. Independently controllable features. *arXiv preprint arXiv:1708.01289*, 2017.

[74] Alan Mathison Turing. The chemical basis of morphogenesis. *Bulletin of mathematical biology*, 52(1-2):153–197, 1990.

[75] Herke Van Hoof, Nutan Chen, Maximilian Karl, Patrick van der Smagt, and Jan Peters. Stable reinforcement learning with autoencoders for tactile and visual data. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3928–3934. IEEE, 2016.

[76] Manuel Watter, Jost Springenberg, Joschka Boedecker, and Martin Riedmiller. Embed to control: A locally linear latent dynamics model for control from raw images. In *Advances in neural information processing systems*, pages 2746–2754, 2015.

[77] Svante Wold, Kim Esbensen, and Paul Geladi. Principal component analysis. *Chemometrics and intelligent laboratory systems*, 2(1-3):37–52, 1987.

[78] Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. How transferable are features in deep neural networks? In *Advances in neural information processing systems*, pages 3320–3328, 2014.

[79] Amy Zhang, Harsh Satija, and Joelle Pineau. Decoupling dynamics and reward for transfer learning. *arXiv preprint arXiv:1804.10689*, 2018.

[80] Ying Zhang, Chunjiang Zhao, Jianjun Du, Xinyu Guo, Wenliang Wen, Shenghao Gu, Jinglu Wang, and Jiangchuan Fan. Crop phenomics: current status and perspectives. *Frontiers in Plant Science*, 10:714, 2019.

Supplementary Material

This supplementary material provides implementation details, hyper-parameters settings, additional results and visualisations.

- Section A presents a focus on the design choices we use for IMGEP-HOLMES
- Section B provides implementation details for the main paper evaluation procedure
  - B.1: Quantitative evaluation of diversity
  - B.2: Quantitative evaluation of Representational Similarity
  - B.3: Human-evaluator selection of the BC spaces for evaluating SLP and TLP diversity
- Section C provides all necessary implementation details for reproducing the main paper experiments
  - C.1: Lenia environment settings
  - C.2: Parameter-sampling policy Π settings for Lenia's initial state and update rule
  - C.3: Settings for training the BC spaces in IMGEP-VAE and IMGEP-HOLMES
- Section D provides additional results that complete the ones from the main paper
  - D.1: Complete RSA analysis of the hierarchy of behavioral characterizations learned in HOLMES
  - D.2: Additional IMGEP baselines with a monolithic BC space are compared
  - D.3: Ablation study of the impact of the lateral connections in HOLMES
- Section E discusses the comparison of HOLMES with other model-expansion architectures
- Section F provides qualitative visualisations of the hierarchical trees that were autonomously constructed by the different IMGEP-HOLMES variants.

Source code: Please refer to the project website `http://mayalenE.github.io/holmes/` for the source code and complete database of discoveries for our experiments.

A Focus on IMGEP-HOLMES

Algorithm 1: IMGEP-HOLMES pseudo-code. Please refer to section 3 of the main paper for the step-by-step implementation choices and to section C in apppendix for the implementation details.

Algorithm 1 IMGEP-HOLMES

Inputs: Parameter-sampling policy Π
Initialize root representation $\mathcal{R} = \{\mathcal{R}_0\}$
for $k \leftarrow 1$ **to** N **do**
  if $k < N_{init}$ **then** *# initial random iterations*
    Sample $\theta \sim \mathcal{U}(\Theta)$
  else *# goal-directed iterations*
    Sample a target BC space $\hat{BC} \sim G_s(\mathcal{H})$
    Sample a goal $\hat{g} \sim G(\hat{BC}, \mathcal{H})$ in $\hat{BC}$
    Choose $\theta \sim \Pi(\hat{BC}, \hat{g}, \mathcal{H})$
  Rollout experiment with θ and observe o

  # Encode reached goals in the hierarchy
  Start with root node $i \leftarrow 0, parent(i) = \emptyset$
  while i exists in the hierarchy (until leaf) **do**
    $r_i = \mathcal{R}_i(o, \mathcal{R}_{parent(i)}(o))$
    Append (θ, o, r_i) to the history $\mathcal{H}$
    $i \leftarrow ic, c = \mathcal{B}_i(r_i)$ *# go to left or right child*

  # Augment representational capacity
  if a BC space BC_i is *saturated* **then**
    Freeze $\mathcal{R}_i$ weights
    Define a boundary $\mathcal{B}_i : BC_i \rightarrow \{0, 1\}$
    Instantiate child modules $\mathcal{R}_{i0}$ and $\mathcal{R}_{i1}$
    # Project past discoveries to children BCs
    for $(\theta, o, r) \in \mathcal{H}[BC_i]$ **do**
      $\mathcal{R}_j \leftarrow \mathcal{R}_{ic}, c = B_i(r)$
      Append $(\theta, o, \mathcal{R}_j(o))$ to $\mathcal{H}[j]$

 # Periodically train HOLMES
if training requested **then**
  for E epochs **do**
    Train the hierarchy $\mathcal{R}$ on observations in $\mathcal{H}$ with importance sampling
  # Update the database of reached goals
  for $i \in$ hierarchy **do**
    for $(\theta, o, r) \in \mathcal{H}[i]$ **do**
      $\mathcal{H}[i][r] \leftarrow \mathcal{R}_i(o)$

 # Ask for user feedback
if user feedback requested **then**
  Ask user to score leaf BC spaces
  Update G_s with assigned scores

A.1 Design choices in HOLMES

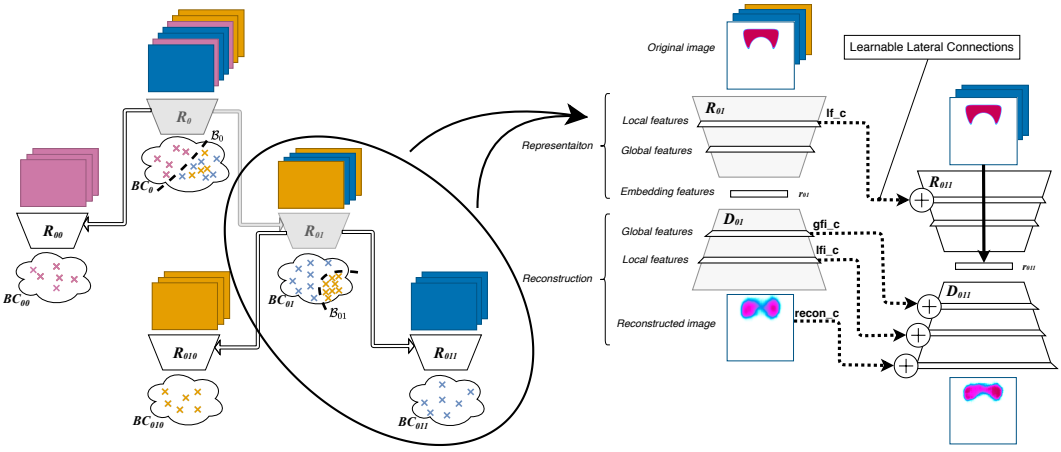


Figure 6: Focus on the different design choices made for the HOLMES architecture. (Left) Each module uses a VAE [33] as the base architecture, where the embedding R_i is coupled to a decoder D_i (D_i is not shown on the left panel for readability). All non-leaf node VAEs are frozen as well as their incoming lateral connections (light grey). The leaf nodes are incrementally trained on their own niches of patterns (represented as colored squares above the embeddings) defined by the boundaries fitted at each node split (curved dotted lines in each BC space, represented as clouds). (Right) R_{011} is trained to encode new information in a latent representation r_{011} (plain vertical arrow) by learning to reuse its parent knowledge via the lateral connections (dotted arrows, denoted as l_f, gfi_c, lfi_c, recon_c).

While the global architecture is generic and numerous design choices can be made, this section details the practical implementation for the *modules*, *connection scheme*, and *splitting criteria* used in this paper. We summarize those components in Figure 6.

Choice for the base module Each module has an embedding network R_i that maps an observation o to a low-dimensional vector $r = R(o)$. To learn such embedding, we rely on a variational autoencoder network [33] for the base module. The encoder network $R_i : q_\phi(r|x)$ is coupled with a decoder network $D_i : p_\theta(x|r)$ that enables a generative process from the latent space, and the networks are jointly trained to maximize the marginal log-likelihood of the training data with a regularizer on the latent structure.
The training loss is $\mathcal{L}_{\text{VAE}}(\theta, \phi; \mathbf{x}, \mathbf{r}) = \underbrace{\mathbb{E}_{\hat{p}(\mathbf{x})}\left(\mathbb{E}_{q_\phi(\mathbf{r}|\mathbf{x})}\left(-\log p_\theta(\mathbf{x}|\mathbf{r})\right)\right)}_{a} + \underbrace{\mathbb{E}_{\hat{p}(\mathbf{x})}\left(D_{\text{KL}}\left(q_\phi(\mathbf{r}|\mathbf{x})||p(\mathbf{r})\right)\right)}_{b}$,
where (a) represents the expected reconstruction error (computed with binary cross entropy) and (b) is the regularizer KL divergence loss of the approximate diagonal Gaussian posterior $q_\phi(r|x)$ from the standard Gaussian prior $p(z) = \mathcal{N}(0, I)$. Please note that input observations are partitioned between the different nodes in HOLMES, therefore each module VAE is trained only on its niche of patterns. Only the encoder network R_i is kept in IMGEP-HOLMES (Algorihm 1), therefore other choices for the base module and training strategy could be envisaged in future work, for instance with contrastive approaches instead of generative approaches.

Choice for the connection scheme The connection scheme takes inspiration from *Progressive Neural Networks* (PNN) [64]), where transfer is enabled by connecting the different modules via learned *lateral connections*. To mitigate the growing number of parameters, we opted for a sparser connection scheme that in [64]. The connection scheme is summarized in Figure 6. Connections are only instantiated between a child and its parent (hierarchical passing of information). Connections are only instantiated between a reduced number of layers (denoted as l_f, gfi_c, lfi_c, recon_c in the figure). We hypothesize that transfer is beneficial in the decoder network so a child module can reconstruct "as well as" its parent, however connections are removed between encoders as new complementary type of features should be learned. We preserve the connections only at the local feature level, as the CNN first layers tend to learn similar features [78]. Connections between linear layers are defined as linear layers and connections between convolutional layers are defined as convolutions with 1×1 kernel. At each connection level, the output of the connection is summed to the current feature map in the VAE. Other connection schemes could be envisaged in future work, for instance with FiLM layers [55] (feature-wise affine transformation instead of sum) which have recently been proposed for vision models.

Choice for the splitting criteria There are two main choices: *when* to split a node and *how* to redirect the patterns toward either the left or right children. For both, we opted for simple design choices that allow the split to be unsupervisedly and autonomously handled during the exploration loop. We trigger a split in a node when the reconstruction loss of its VAE reaches a plateau, with additional conditions to prevent premature splitting (minimal node population and minimal number of training steps) or to limit the total number of splits. When splitting a node, we use K-means algorithm in the embedding space to fit 2 clusters on the points that are currently in the node. This generates a boundary in the latent space of the node, that we keep fixed for the rest of the exploration loop. Again, many other choices could be envisaged in future work, for instance by including human feedback to fit the boundary or with more advanced clustering algorithms.

B Complete Description of the Evaluation Procedure

B.1 Evaluation of diversity

B.1.1 Construction of 5 analytic BC spaces

This section details the 5 BC spaces introduced in section 4.1 of the main paper. Each set of BC features relies either on *engineered* representation based on existing image descriptors from the literature or on *pretrained* representations unsupervisedly learned on Lenia patterns. Those BCs were constructed to characterize different *types* of diversities in the scope of evaluating *meta*-diversity as defined in section 2, but obviously many others could be envisaged. The 5 BC models are provided with the source code of this paper.

Each set of BC features is defined by a mapping function $BC_X : o \in [0,1]^{256\times256} \mapsto \hat{z} \in [0,1]^8$ where X is the corresponding BC space, o is a Lenia pattern and $\hat{z}$ represents its 8-dimensional behavioral descriptor in the corresponding BC space.

We denote $\mathcal{D}_{ref}$ an external dataset of 15000 Lenia patterns. The patterns in $\mathcal{D}_{ref}$ were randomly collected from prior exploration experiments in Lenia, experiments that include different random seeds and different exploration variants and comport 50% SLPs and 50% TLPs. $\mathcal{D}_{ref}$ is a large database that is intended to cover a diversity of patterns orders of magnitude larger than what could be found in any single algorithm experiment, and that we use as reference dataset to construct and normalize the different evaluation BC spaces.

Spectrum-Fourier The 2-dimensional discrete Fourier transform is a mathematical method that projects an image (2D spatial signal) into the frequency domain, from which frequency characteristics can be extracted and used as texture descriptors [70]. Applications range from material description [50], leaf texture description in biology [13] and rule classification in cellular automata [47].

The construction of $\text{BC}_{\text{SPECTRUM-FOURIER}}$ is summarized in Figure 7 and follows the below procedure:

1. The 2D Fast Fourier Transform transforms the image $o = f(x,y)$ into the u,v frequency domain function F, the zero-frequency component is shifted to the center of the array and the power specrum PS (or power spectral density) is computed:

$$F(u,v) = \frac{1}{256\times256}\sum_{x=0}^{255}\sum_{y=0}^{255} f(x,y)\exp^{-j2\pi\frac{ux}{256}\frac{vx}{256}}$$
$$F(u,v) \leftarrow Roll(F(u,v),(\frac{256}{2},\frac{256}{2}))$$
$$PS(u,v) = Real(F(u,v))^2 + Imaginary(F(u,v))^2$$

2. The power spectrum is filtered to keep only the lower half (symmetry property of the FFT) and the significant values:

$$PS(u,v) = \{PS(u,v), 0 \le u \le \frac{256}{2}, -\frac{256}{2} \le v \le \frac{256}{2} - 1\}$$
$$PS(u,v) = 0\, if\ PS(u,v) < mean(PS(u,v))$$

3. The power spectrum is partitioned into 20 ring-shaped sectors:

$$\left[R_i = \{PS(u,v) | r_1^2 \le u^2 + v^2 \le r_2^2\}\ with\ (r_1,r_2) = (\frac{i}{20}\times\frac{256}{2}, \frac{i+1}{20}\times\frac{256}{2});\ for\ i \in [0..19]\right]$$

4. A 40-dimensional feature vector (FV) representing radially-aggregating measures (mean μ_i and standard deviation σ_i of each sector) is extracted:

$$FV(o) = [\mu_1,\sigma_1,\ldots,\mu_{20},\sigma_{20}],$$
$$\text{where } \mu_i = mean(PS[R_i]), \sigma_i = std(PS[R_i])$$

5. The 40-dimensional feature vector FV is projected into a normalized 8-dimensional behavioral descriptor $\hat{z}$ using a transformation $\hat{T} : FV \mapsto \hat{z}$. $\hat{T}$ is constructed with Principal

Component Analysis (PCA) [77] dimensionality reduction on $\mathcal{D}_{ref}$:

$X_{ref} = \{FV(o), o \in \mathcal{D}_{ref}\}$

Fit a PCA with 8 components on X_{ref}, $PCA : FV \in \mathbb{R}^{40} \mapsto z \in \mathbb{R}^8$

$z_{ref} = PCA(X_{ref}), z_{min} = percentile(z_{ref}, 0.01), z_{max} = percentile(z_{ref}, 99.9))$

$$\hat{T} : FV \mapsto \hat{z} = \frac{PCA(FV) - z_{min}}{z_{max} - z_{min}}$$

6. $\text{BC}_{\text{SPECTRUM-FOURIER}}(o) = \hat{T} \circ FV(o)$

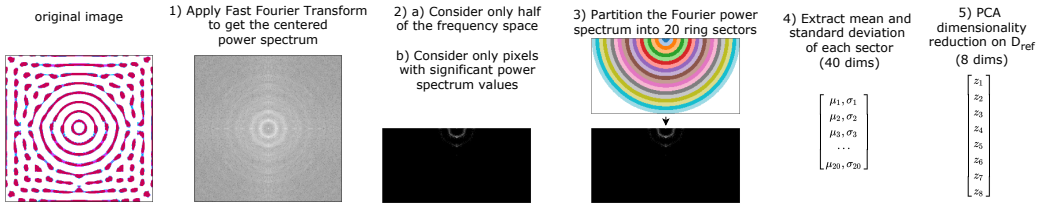


Figure 7: Construction of SPECTRUM-FOURIER analytic space. See text for details. Please note that for visualisation purposes: (left) the original image is colorized but is originally a 256×256 grayscale image; (step 1-2-3) the power spectrum is depicted in logarithmic scale.

Elliptical-Fourier Elliptical Fourier analysis (EFA) [36] is a mathematical method for contour description which has been widely-used for shape description in image processing [63]. These descriptors have been applied to morphometrical analysis in biology [44], for instance to characterize the phenotype of plants leaf and petal contours [52] or anatomical shape changes [10, 22].

A closed contour $\{x_p, y_p\}_{p=1}^{K}$ (K points polygon) can be seen as a continuous periodic function of the *length* parameter $T = \sum_{p=1}^{K} \Delta t_p$ where t_p is the distance from the $p-1^{th}$ to the p^{th} point. Therefore it can be represented as a sum of cosine and sine functions of growing frequencies (harmonics) under Fourier approximation. Each harmonic is an ellipse which is defined by 4 coefficients a, b, c, d.

The construction of $\text{BC}_{\text{ELLIPTICAL-FOURIER}}$ is summarized in Figure 8 and follows the below procedure:

1. Binarize the image $o_{\texttt{binary}}$ `= o > 0.2` and extract the external contour as the a list of the (x,y) positions of the pixels that make up the boundary using OpenCV function `contour = cv2.findContours(`$o_{\texttt{binary}}$`, cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_SIMPLE)`
2. Extract the set of $\{a_n, b_n, c_n, d_n\}_{n=1}^{N}$ coefficients for a series of N ellipses (N=25) from the x- and y-deltas (Δx_p and Δy_p) between each consecutive point p in the K points polygon:

$$a_n = \frac{T}{2n^2\pi^2} \sum_{p=1}^{K} \frac{\Delta x_p}{\Delta t_p} \left[\cos \frac{2n\pi t_p}{T} - \cos \frac{2n\pi t_{p-1}}{T} \right]$$

$$b_n = \frac{T}{2n^2\pi^2} \sum_{p=1}^{K} \frac{\Delta x_p}{\Delta t_p} \left[\sin \frac{2n\pi t_p}{T} - \sin \frac{2n\pi t_{p-1}}{T} \right]$$

$$c_n = \frac{T}{2n^2\pi^2} \sum_{p=1}^{K} \frac{\Delta y_p}{\Delta t_p} \left[\cos \frac{2n\pi t_p}{T} - \cos \frac{2n\pi t_{p-1}}{T} \right]$$

$$d_n = \frac{T}{2n^2\pi^2} \sum_{p=1}^{K} \frac{\Delta y_p}{\Delta t_p} \left[\sin \frac{2n\pi t_p}{T} - \sin \frac{2n\pi t_{p-1}}{T} \right]$$

3. The coefficients are standardized (i.e. made invariant to size, rotation and shift): $\begin{bmatrix} a_n^* & b_n^* \\ c_n^* & d_n^* \end{bmatrix} = \frac{1}{L} \begin{bmatrix} \cos\phi & \sin\phi \\ -\sin\phi & \cos\phi \end{bmatrix} \begin{bmatrix} a_n & b_n \\ c_n & d_n \end{bmatrix} \begin{bmatrix} \cos N\theta & \sin N\theta \\ -\sin N\theta & \cos N\theta \end{bmatrix}$, where $L = \sqrt{[(A_0 - x_m)^2 + (C_0 - x_m)^2]}$, (A_0, C_0) is the center of the 1^{st} harmonic ellipse, (x_m, y_m)

is the location of the modified starting point (on the major axis of the ellipse), $\theta = \frac{2\pi t_m}{T}$ and $\phi = \tan^{-1} \frac{y_m - C_0}{x_m - A_0}$ (angle between the major axis of the ellipse and xaxis).

4. The 100-dimensional feature vector $FV = \{a_n^*, b_n^*, c_n^*, d_n^*\}_{n=1}^{25}$ is projected into a normalized 8-dimensional behavioral descriptor using a transformation $\hat{T} : FV \mapsto \hat{z}$. $\hat{T}$ is constructed with Principal Component Analysis (PCA) dimensionality reduction on $\mathcal{D}_{ref}$ (similar procedure as in point 5 of $\text{BC}_{\text{SPECTRUM-FOURIER}}$).
5. $\text{BC}_{\text{ELLIPTICAL-FOURIER}}(o) = \hat{T} \circ FV(o)$

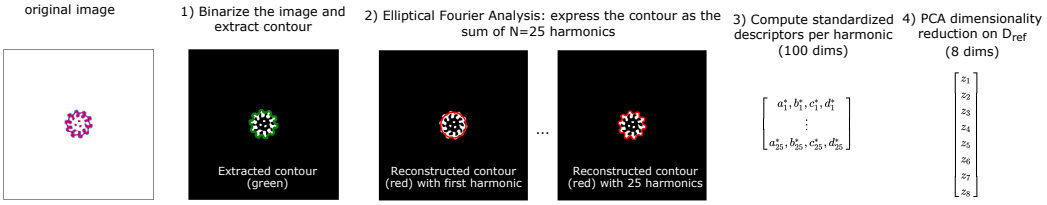


Figure 8: Construction of ELLIPTICAL-FOURIER analytic space. See text for details. (step 1) The contour depicted in green is extracted with OpenCV `findContours()` function (step 2) The contours depicted in red are reconstructed from the EFA coefficients (like in other Fourier series transforms the shape signal can be approximated by summing the harmonics [36]).

Lenia-Statistics The original Lenia paper proposes several measures for statistical analysis of the Lenia patterns (section 2.4.2 in [6]), also defined in Reinke et al. (2020) (section B.3 in [62]).

$\text{BC}_{\text{LENIA-STATISTICS}}$ is constructed on top of these measures according to the below procedure:

1. Among all the statistical measures proposed in [6] we selected the 17 measures that are time-independent, i.e. that can be computed from the final Lenia pattern $o = I(x, y)$, namely:
   - the activation mass $m = \frac{1}{256 \times 256} \sum_{(x,y) \in I} I(x, y)$
   - the activation volume $V_m = \frac{1}{256 \times 256} \sum_{(x,y) \in I} \delta_{I(x,y) > \epsilon}$ ($\epsilon = 10^{-4}$)
   - the activation density $\rho_m = \frac{m}{V_m}$
   - the centeredness of the activation mass distribution $C_m = \frac{1}{m} \sum_{(x,y) \in I} w_{xy} \cdot I(x - \bar{x}_m, y - \bar{y}_m)$ where $(\bar{x}_m, \bar{y}_m)$ is the activation centroid and $w_{x,y} = \left(1 - \frac{d(x,y)}{\max_{x,y} d(x,y)}\right)^2$ with $d(x, y) = \sqrt{(x - \bar{x}_m)^2 + (y - \bar{y}_m)^2}$
   - the 8 invariant image moments by Hu [30]
   - the 5 extra invariant image moments by Flusser [20]
2. The 17-dimensional feature vector $FV = [m, V_m, \rho_m, C_m, hu_1, \ldots, hu_7, flusser_8, \ldots, flusser_{13}]$ is projected into a normalized 8-dimensional behavioral descriptor using a transformation $\hat{T} : FV \mapsto \hat{z}$. $\hat{T}$ is constructed with Principal Component Analysis (PCA) dimensionality reduction on $\mathcal{D}_{ref}$ (similarly than for $\text{BC}_{\text{SPECTRUM-FOURIER}}$).
3. $\text{BC}_{\text{LENIA-STATISTICS}}(o) = \hat{T} \circ FV(o)$

BetaVAE Reinke et al. (2020) [62] propose to train a β-VAE [5] on a large database of Lenia pattterns and to reuse the learned features as behavioral descriptors for the analytic BC space.

$\text{BC}_{\text{BETAVAE}}$ is constructed according to the below procedure :

1. A β-VAE with 8-dimensional latent space is instantiated with the architecture detailed in table 1.
2. The construction of the training dataset, training procedure and hyperparameters follows [62]:

- The β-VAE is trained on an external database $\mathcal{D}_{ref}^{(big)}$ of 42500 Lenia patterns (with 50% SLP and 50% TLP, 37500 as training set, 5000 as validation set) which were randomly collected from independent previous experiments (with the same procedure than $\mathcal{D}_{ref}$).
- The β-VAE is trained for more than 1250 epochs with hyperparameters $\beta = 5$, Adam optimizer ($lr = 1\text{e}{-3}$, $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon = 1\text{e}{-8}$, weight decay=1e−5) and a batch size of 64.
- The network weights which resulted in the minimal validation set error during the training are kept.

3. The resulting pretrained encoder serves as mapping function from a Lenia pattern o to a 8-dimensional feature vector $FV(o) = [z_1, z_2, z_3, z_4, z_5, z_6, z_7, z_8]$
4. Similarly to the other analytic BC spaces in this paper, we use the reference dataset $\mathcal{D}_{ref}$ to normalize the 8-dimensional behavioral descriptors between $[0, 1]$:

   $z_{ref} = \{FV(o), o \in \mathcal{D}_{ref}\}, z_{min} = percentile(z_{ref}, 0.01), z_{max} = percentile(z_{ref}, 99.9))$

   $\hat{T} : FV \mapsto \hat{z} = \dfrac{FV - z_{min}}{z_{max} - z_{min}}$
5. $\text{BC}_{\text{BETAVAE}}(o) = \hat{T} \circ FV(o)$

Patch-BetaVAE Reinke et al. (2020) noticed that the β-VAE is not able to encode finer details and texture of patterns as the compression of the images to a 8-dimensional vector results in a general blurriness in the reconstructed patterns [62]. Therefore, we also implemented an additional variant denoted as PATCH-BETAVAE where the β-VAE is trained on "zoomed" 32×32 patches. A preprocessing step extracts the cropped patch around the image activation centroid $P : o \mapsto o[\bar{x}_m - 16 : \bar{x}_m + 16, \bar{y}_m - 16 : \bar{y}_m + 16]$. Then, the construction of $\text{BC}_{\text{PATCH-BETAVAE}}$ follows exactly the construction of $\text{BC}_{\text{BETAVAE}}$, except that the network architecture has only 3 convolutional layers instead of 6. Following the notations of the previous paragraph, $\text{BC}_{\text{PATCH-BETAVAE}}(o) = \hat{T} \circ FV \circ P(o)$ with FV the pretrained model on image patches and $\hat{T}$ a normalizing function computed on $\mathcal{D}_{ref}$.

B.1.2 Binning-based Diversity Metric

We follow existing approaches in the QD and IMGEP literature [55, 57] and use binning-based measure to quantify the diversity of a set of explored instances into a predefined BC space. The entire BC space is discretized into a collection of t bins $N_1, \ldots, N_t$ and the diversity is quantified as the number of bins filled over the course of exploration: $D_{|BC} = \sum_{i=1}^{t} \delta_i$ where $\delta_i = 1$ if the N_i^{th} bin is filled, $\delta_i = 0$ otherwise.

We opt for a regular binning where each dimension of the BC space is discretized into equally sized bins. For all the results in the main paper, 20 bins per dimension are used for the discretization of the BC spaces. For recall, all the analytic BC spaces used in this paper are 8-dimensional and bounded in $[0, 1]^8$ (see previous section). This results in a total of 25.6×10^9 bins. Note however that for a given BC space, the maximum number of bins that can be filled by all possible Lenia patterns is unknown.

Because binning-based metrics directly depend on the choice of the bins discretization, we analyze in Figure 9 the impact of the choice of the number of bins on the final diversity measure. As we can see,

Encoder	**Decoder**
Input pattern A: $256 \times 256 \times 1$	Input latent vector z: 8×1
Conv layer: 32 kernels 4×4, stride 2, 1-padding + ReLU	FC layers : 256 + ReLU, 256 + ReLU, $4 \times 4 \times 32$ + ReLU
Conv layer: 32 kernels 4×4, stride 2, 1-padding + ReLU	TransposeConv layer: 32 kernels 4×4, stride 2, 1-padding + ReLU
Conv layer: 32 kernels 4×4, stride 2, 1-padding + ReLU	TransposeConv layer: 32 kernels 4×4, stride 2, 1-padding + ReLU
Conv layer: 32 kernels 4×4, stride 2, 1-padding + ReLU	TransposeConv layer: 32 kernels 4×4, stride 2, 1-padding + ReLU
Conv layer: 32 kernels 4×4, stride 2, 1-padding + ReLU	TransposeConv layer: 32 kernels 4×4, stride 2, 1-padding + ReLU
Conv layer: 32 kernels 4×4, stride 2, 1-padding + ReLU	TransposeConv layer: 32 kernels 4×4, stride 2, 1-padding + ReLU
FC layers : 256 + ReLU, 256 + ReLU, FC: 2×8	TransposeConv layer: 32 kernels 4×4, stride 2, 1-padding

Table 1: β-VAE architecture used for $\text{BC}_{\text{BETAVAE}}$.

the ranking of the different IMGEP algorithms compared in Figure 5 of the main paper is invariant to this choice.

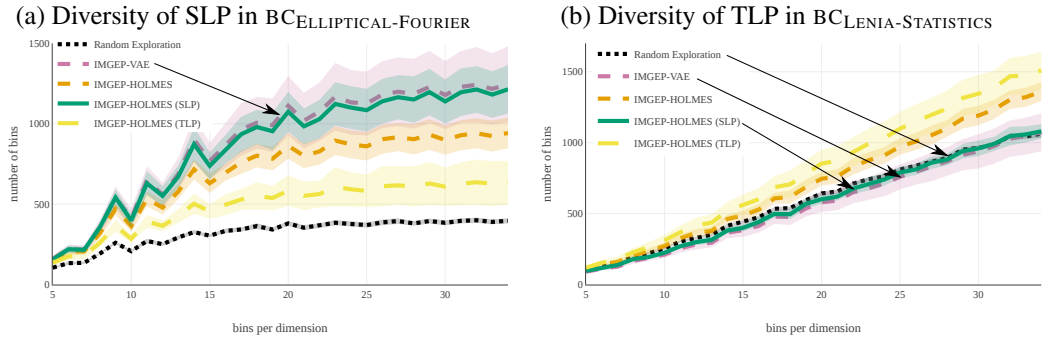


Figure 9: Influence of the choice of the number of bins on the diversity measure presented in Figure 5 of the main paper. The final diversity (number of occupied bins at the end of exploration, as shown in the y axis) is measured by varying the number of bins per dimension from 5 to 35. Results in the main paper use n=20 bins. Mean and std-deviation shaded area curves are depicted.

B.2 Representational Similarity Analysis

We denote $\mathcal{D}_{ref}^{(small)}$ an external dataset of 3000 Lenia patterns (50% SLPs and 50% TLPs) which were collected with the same procedure than $\mathcal{D}_{ref}$.

Given two representations embedding networks R_i and R_j with 8-dimensional latent space, the RSA similarity index RSA_{ij} is computed with the the linear Centered Kernel Alignment index (CKA) as proposed in [34]:

1. Compute the matrix of behavioral descriptors responses from each representation $Z_i = [R_i(o), o \in \mathcal{D}_{ref}^{(small)}] \in [0,1]^{3000\times 8}$ and $Z_j = [R_j(o), o \in \mathcal{D}_{ref}^{(small)}] \in [0,1]^{3000\times 8}$
2. Center the matrices responses: $Z_i \leftarrow Z_i - mean(Z_i, axis = 0)$ and $Z_j \leftarrow Z_j - mean(Z_j, axis = 0)$
3. $RSA_{ij} = CKA(Z_i Z_i^T, Z_j Z_j^T) = \frac{||Z_i \cdot Z_j^T||_F^2}{||Z_i \cdot Z_i^T||_F ||Z_j \cdot Z_j^T||_F}$ where $||\cdot||_F$ represents the Frobenius norm

Representation Similarity Analysis (RSA) is used in Figure 3 of the main paper in two ways:

- To compare representations in *time*, i.e. where the the embedding networks R_i and R_j come from the same network but from different training stages
- To compare representations from different *modules* in HOLMES where the the embedding networks R_i and R_j are taken from the same time step (end of exploration) but from different networks.

B.3 Human-Evaluator Selection of a *proxy*-BC for Evaluation of SLP and TLP Diversity

Relying on the external database $\mathcal{D}_{ref}$ of 15000 Lenia patterns (50% SLP - 50% TLP) and the SLP/TLP classifiers, we conducted an experiment with a human evaluator to select the analytic BC space that correlates the most with human judgement of what represents a *diversity of SLP* and a *diversity of TLP*.

The experiment consisted in repeatedly showing the human with two sets of patterns (as shown in Figure 10) and asking the human to click on the set that he considers is the more *diverse*, according to its intuitive notion of diversity. If the human cannot choose between the two sets, he can click on the "pass" button. In background, the procedure to generate the sets is the following:

1. Randomly select a (BC, category) pair, where BC $\in$ {SPECTRUM-FOURIER, ELLIPTICAL-FOURIER, LENIA-STATISTICS, BETAVAE, PATCH-BETAVAE} and category $\in$ {SLP,TLP}.

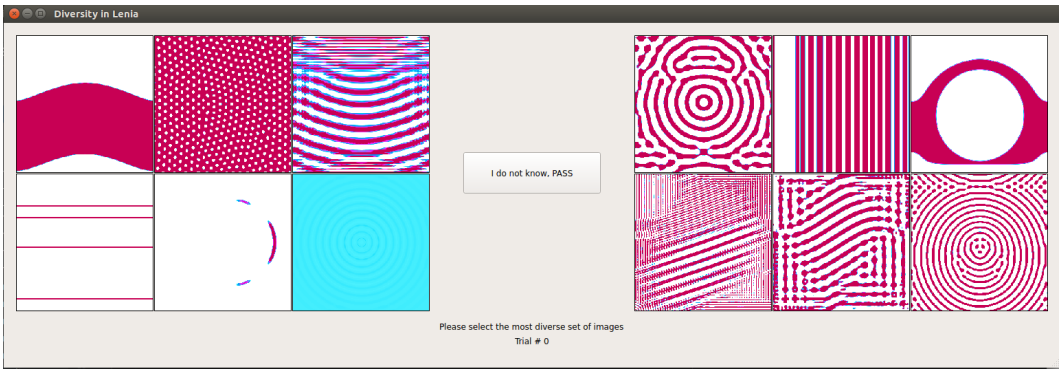


Figure 10: Interface used for selecting of the best proxy-BC analytic space that correlates with human judgement of what represents a *diversity of SLP* and a *diversity of TLP*.

2. Randomly draw 750 candidate sets of 6 images among the 7500 patterns of the current category.
3. Select the most *similar* set and the most *dissimilar* (i.e. diverse) set among those 750 sets. The (di-)ssimilarity of a set of 6 images is measured as a function of all the distances between each pair of images in the set, with distances being computed in the current BC space. This distance-based measure of diversity D, proposed in [65], measures the magnitude M (dispersion) and variability E (equability) of the set of S=6 points in the BC [65]:

$$M = \frac{S}{S-1}\sum_{i=1}^{S}\sum_{j=1}^{S}\frac{d_{ij}}{S^2}, \text{ where } d_{ij} \text{ is the pairwise euclidean distance}$$

$$E = \frac{1+\sqrt{1+4H}}{2S}, \text{ where } H = \left[\sum_{i=1}^{S}\sum_{j=1}^{S}\left(\frac{d_{ij}}{\sum_{i=1}^{S}\sum_{j=1}^{S}d_{ij}}\right)^2\right]^{\frac{1}{1-2}}$$

$$D = 1 + (S-1) \times E \times M, M \in [0,1] and E \in [0,1]$$

   This measure replaces the binning-based measure which can hardly be used here (as they are only 6 images most candidate sets are likely to fall in the same number of bins and be equally diverse).
4. The sets are displayed to the human in random presentation order.

This experiment was conducted with one human evaluator which performed a total of 500 clicks, i.e 50 times per (BC, category) pair. For each click per (BC,category) pair, the agreement score is 0 if the human selected the opposite set that the one considered as diverse by the BC, 0.5 is the human selected the "pass" button and 1 if the human selected the set considered as diverse by the BC. Table 2 reports the mean and standard deviation agreement scores of the human evaluator for each (BC, category) pair. The agreement score is significant at level $\alpha = 5\%$ if it is above $0.64 = 0.5 + 1.96 \times \sqrt{\frac{0.25}{50}}$.

Table 2: Human-evaluator agreement scores (mean $\pm$ std). Best scores are shown in bold.

	Spectrum-Fourier	**Elliptical-Fourier**	**Lenia-Statistics**	**BetaVAE**	**Patch-BetaVAE**
SLP	0.5 ± 0.18	$\mathbf{0.98 \pm 0.04}$	0.59 ± 0.12	0.1 ± 0.06	0.89 ± 0.08
TLP	0.2 ± 0.13	0.47 ± 0.1	$\mathbf{0.92 \pm 0.07}$	0.75 ± 0.08	0.38 ± 0.08

As we can see, the human evaluator designated $\text{BC}_{\text{ELLIPTICAL-FOURIER}}$ as the best proxy space for evaluating the diversity of SLP (98% agreement score) and $\text{BC}_{\text{LENIA-STATISTICS}}$ as the best proxy space for evaluating the diversity TLP (92% agreement score). This is why those BCs are used in Figure 5 of the main paper.

C Experimental Settings

C.1 Environment Settings

All experiments are done in the Lenia environment, as described in [6, 62]. As stated in the main paper, we use a 256×256 state size ($A \in \mathbb{R}^{256\times256}$) and a number of $T = 200$ steps for each run.

The 256×256 Lenia grid is a torus where the neighborhood is circular (i.e pixels on the top border are neighbors of the pixels on the bottom border and same between the left and right borders).

Lenia's update rule ($A^t \rightarrow A^{t+1}$) is defined as $A^{t+1} = A^t + \Delta_{\mathcal{T}} G(K * A^t)$, where:

- G defines a parametrized *growth mapping* function: exponential with $G(u; \mu, \sigma) = 2\exp\left(-\frac{(u-\mu)^2}{2\sigma^2}\right) - 1$
- K defines a parametrized concentric-ring Kernel with:

$$K_C(r) = \exp\left(\alpha - \frac{\alpha}{4r(1-r)}\right), \text{ with } \alpha = 4 \text{ (Kernel core)}$$
$$K_S(r;\beta) = \beta_{\lfloor Br \rfloor} K_C(Br \ mod \ 1), \text{ with } \beta = (\beta_1, \beta_2, \beta_3) \text{ (Kernel shell)}$$
$$K = \frac{K_S}{|K_S|}$$

The update rule is therefore determined with 7 parameters:

- R: radius of the Kernel (i.e. radius of the local neighborhood that influences the evolution of each cell in the grid),
- $\mathcal{T} = \frac{1}{\Delta_{\mathcal{T}}}$: fraction of the growth update that is applied per time step,
- μ, σ: growth center and growth width,
- $\beta_1, \beta_2, \beta_3$: concentring rings parameters that control the shape of the kernel.

See the project website `http://mayalenE.github.io/holmes/` for videos of the Lenia dynamics.

C.2 Sampling of parameters θ

The set of *controllable* parameters θ of the artificial agent include:

- The update rules parameters $[R, \mathcal{T}, \mu, \sigma, \beta_1, \beta_2, \beta_3]$
- CPPN-parameters that control the generation of the initial state $A^{t=1}$

For each exploration run, the IMGEP agent samples a set of parameters $\theta \in \Theta$ that generates a rollout $A^{t=1} \rightarrow \cdots \rightarrow A^{t=200}$.

Following the notations of Algorithm 1, there are two ways parameters $\theta \in \Theta$ are sampled:

1. During the N_{init} initial runs, parameters are randomly sampled $\theta \sim \mathcal{U}(\Theta)$
2. During the goal-directed exploration runs, parameters are sampled from a policy $\theta \sim \Pi(\hat{BC}_i, \hat{g}, \mathcal{H})$. The Π policy operates in two steps:
   (a) given a goal $g \in \hat{BC}_i$, select parameters $\hat{\theta} \in \mathcal{H}$ whose corresponding outcome is closest to g in $\hat{BC}_i$
   (b) mutate the parameters by a random process $\theta = \text{MUTATION}(\hat{\theta})$

We therefore need to define 1) the random process $\mathcal{U}$ used to randomly initialize the parameters θ and 2) the random MUTATION process used to mutate an existing set of parameters $\hat{\theta}$.

Please note that for both we follow exactly the implementation proposed in Reinke et al. (2020) [62]. We therefore refer to section B.4 of their paper for a complete description of the implementation of the random initialization process and random mutation process. This includes the procedure used

for parameters that control the generation of the initial pattern $A^{t=1}$ and for parameters that control Lenia's update rule.

In this paper, we use the exact same hyperparameters as in [62] for initialization $\mathcal{U}$ and MUTATION of the CPPN-parameters that control the generation of the initial state $A^{t=1}$. We use slightly different hyper-parameters for the MUTATION of the parameters that control the generation of the update rule $[R, \mathcal{T}, \mu, \sigma, \beta_1, \beta_2, \beta_3]$, as detailed in table 3.

Table 3: Sampling of parameters for the update rule. The random initialization process $\mathcal{U}$ uses uniform sampling in an interval $[a, b]$. The random MUTATION is a Gaussian process $\theta = [\hat{\theta} + \mathcal{N}(\sigma_M)]_a^b$.

	R	$\mathcal{T}$	μ	σ	$(\beta_1, \beta_2, \beta_3)$
$[a, b]$	$[2, 20]$	$[1, 20]$	$[0, 1]$	$[0.001, 0.3]$	$[0, 1]$
σ_M	0.5	0.5	0.1	0.05	0.1

C.3 Incremental Training of the BC Spaces

Training Procedure The networks are trained 100 epochs every 100 runs of exploration (resulting in 50 training stages and 5000 training epochs in total). The networks are initialized with *kaiming* uniform initialization. We used the Adam optimizer ($lr = 1\text{e}{-3}$, $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon = 1\text{e}{-8}$, weight decay=$1\text{e}{-5}$) with a batch size of 128.

Training Dataset The datasets are incrementally constructed during exploration by gathering the discovered patterns. One pattern every ten is added to the validation set (10%) and the rest is used in the training set (the validation dataset only serves for checking purposes and has no influence on the learned BC spaces). Importance sampling is used to give the newly-discovered patterns more weights. A weighted random sampler is used as follow: at each training stage t, there are X patterns discovered so far among which X_{new} have been discovered during the last 100 steps, we create a dataset D_t of X images that we construct by sampling 30% among the X_{new} lastly discovered images and 70% among the $X - X_{new}$ old patterns. We also use data-augmentation, i.e at each training stage t, the images in D_t are augmented online by random x and y translations (up to half the pattern size and with probability 0.6), rotation (up to 20 degrees and with probability 0.6), horizontal and vertical flipping (with probability 0.2), zooming (up to factor 3 with probability 0.6). The augmentations are preceded by spherical padding to preserve Lenia spherical continuity.

IMGEP-VAE The monolithic VAE architecture used in the IMGEP-VAE baseline is detailed in table 4. It has a total neural capacity of 2258657 parameters.

Encoder	Decoder
Input pattern A: $256 \times 256 \times 1$	Input latent vector z: 16×1
Conv layer: 64 kernels 4×4, stride 2, 1-padding + ReLU	FC layers : 512+ ReLU, 512+ ReLU, $4 \times 4 \times 64$ + ReLU
Conv layer: 64 kernels 4×4, stride 2, 1-padding + ReLU	TransposeConv layer: 64 kernels 4×4, stride 2, 1-padding + ReLU
Conv layer: 64 kernels 4×4, stride 2, 1-padding + ReLU	TransposeConv layer: 64 kernels 4×4, stride 2, 1-padding + ReLU
Conv layer: 64 kernels 4×4, stride 2, 1-padding + ReLU	TransposeConv layer: 64 kernels 4×4, stride 2, 1-padding + ReLU
Conv layer: 64 kernels 4×4, stride 2, 1-padding + ReLU	TransposeConv layer: 64 kernels 4×4, stride 2, 1-padding + ReLU
Conv layer: 64 kernels 4×4, stride 2, 1-padding + ReLU	TransposeConv layer: 64 kernels 4×4, stride 2, 1-padding + ReLU
FC layers : 512+ ReLU, 512+ ReLU, FC: 2×16	TransposeConv layer: 1 kernels 4×4, stride 2, 1-padding

Table 4: VAE architecture used for IMGEP-VAE.

IMGEP-HOLMES For the IMGEP-HOLMES variant, the hierarchical representation starts with a single root module R_0 at the beginning of exploration. During each training stage, one node is *split* if it meets the following conditions:

- the reconstruction loss for that node reaches a plateau (running average over the last 50 training epochs is below $\epsilon = 20$)
- at least 500 patterns populate the node

- the node has not just been created (must has been trained for at least 200 epochs)
- it is not too early in the exploration loop (there must be at least 2000 patterns are explored)
- the total number of nodes in the hierarchy is below the maximum number allowed (we stop the expansion after 11 splits i.e. 23 modules)

Each time a split is triggered in a BC space node of the hierarchy BC_i, the boundary $\mathcal{B}_i$ is fitted in the latent space as follows: K-Means algorithm with 2 clusters is ran on the patterns that currently populate the node. The resulting clusters are kept fixed for the rest of the exploration, therefore when a pattern is projected in the split node, it is sent to the left children if it belongs to the first cluster on the latent space and to the right children otherwise.
For the IMGEP-HOLMES variant, the final hierarchy has a total of 23 VAE modules. The architecture is identical for each module and is detailed in table 5. At the end of exploration, HOLMES has a total neural capacity of 2085981 parameters. Each base module VAE has a capacity of 86225 parameters and connections of 4673 parameters ($2085981 = 23 \times 86225 + 22 \times 4673$).

Encoder	
Input pattern A: $256 \times 256 \times 1$	
Conv layer: 16 kernels 4×4, stride 2, 1-padding + ReLU	
Conv layer: 16 kernels 4×4, stride 2, 1-padding + ReLU	
Conv layer: 16 kernels 4×4, stride 2, 1-padding + ReLU	**lf_c:** 16 kernels 1×1, stride 1, 1-padding
Conv layer: 16 kernels 4×4, stride 2, 1-padding + ReLU	
Conv layer: 16 kernels 4×4, stride 2, 1-padding + ReLU	
Conv layer: 16 kernels 4×4, stride 2, 1-padding + ReLU	
FC layers : 64+ ReLU, 64+ ReLU, FC: 2×16	
Decoder	
Input latent vector z: 16×1	
FC layers : 64+ ReLU,	**gfi_c:** 64+ReLU
FC layers: 64+ ReLU, $4 \times 4 \times 16$ + ReLU	
TransposeConv layer: 16 kernels 4×4, stride 2, 1-padding + ReLU	
TransposeConv layer: 16 kernels 4×4, stride 2, 1-padding + ReLU	
TransposeConv layer: 16 kernels 4×4, stride 2, 1-padding + ReLU	**lfi_c:** 16 kernels 1×1, stride 1, 1-padding
TransposeConv layer: 16 kernels 4×4, stride 2, 1-padding + ReLU	
TransposeConv layer: 16 kernels 4×4, stride 2, 1-padding + ReLU	
TransposeConv layer: 1 kernel 4×4, stride 2, 1-padding	**recon_c:** 1 kernel 1×1, stride 1, 1-padding

Table 5: Module architecture used for IMGEP-HOLMES. All the modules R_i have this architecture for the base VAE network as well as the connections (except R_0 which does not have the connections).

D Additional Results

D.1 RSA complete temporal analysis and statistics

Figure 11 shows how HOLMES is able to progressively builds a hierarchy of behavioral characterization spaces from the incoming data. The data is here collected by the IMGEP-HOLMES algorithm, with 50 training stages occurring each 100 steps. At start (stage 0), the hierarchy contains only the root node at the top of the figure (BC 0). Node saturation occurs at training stages where the RSA similarity index between the representation at that stage and the representations at all subsequent stages is high (yellow). For example, we see on the figure that the root node saturates after approximately 15-20 training stages. When a node saturates, HOLMES splits it in two child nodes (see section A.1 for details on the splitting procedure). For example, the root node BC 0 is split into the child nodes BC 00 and BC 01 at training stage 21, as indicated by the fact that the RSA plots of

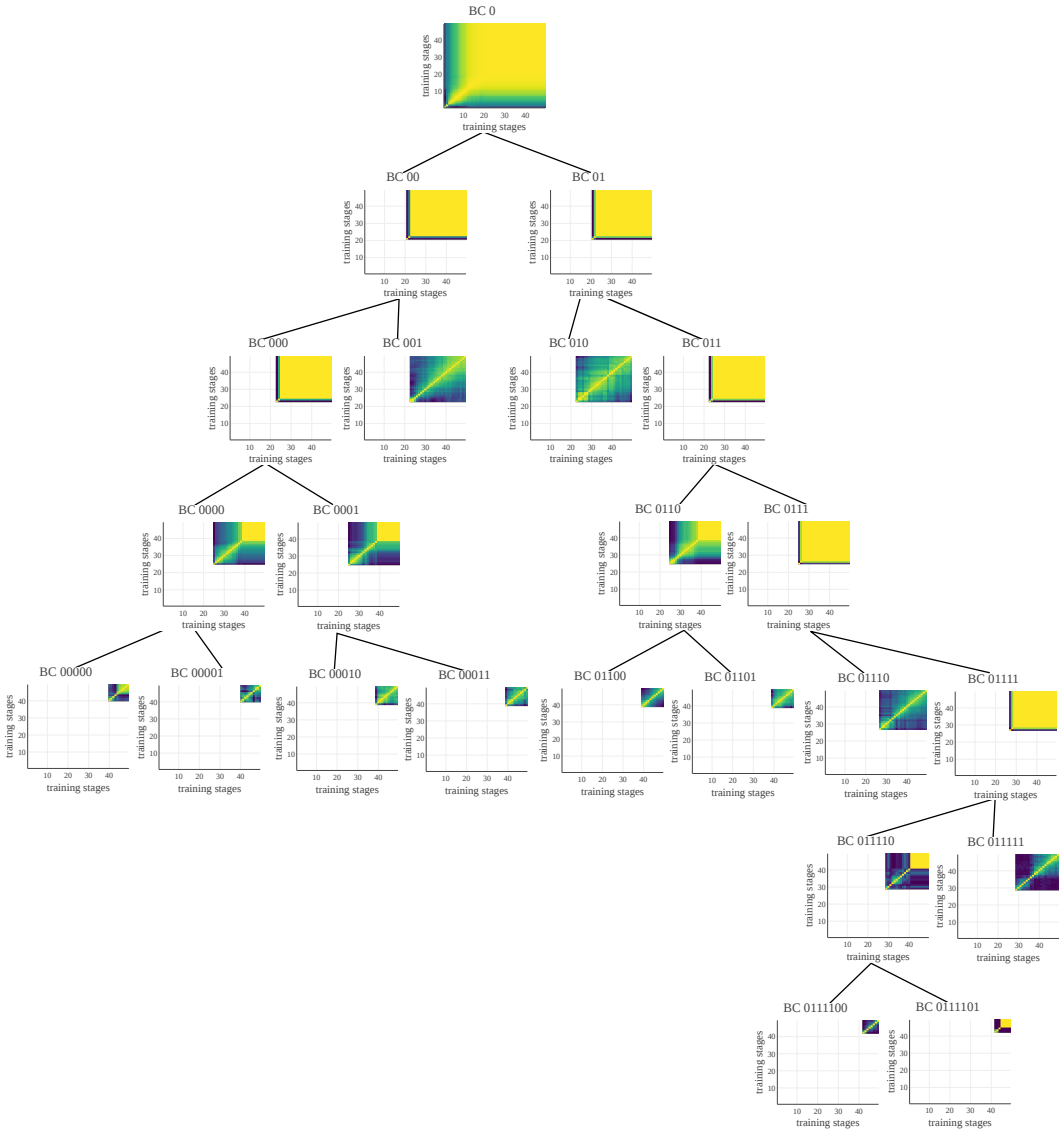


Figure 11: Example of a hierarchy of behavioral characterization spaces learned by HOLMES. It starts with a root node (BC 0, top) and iteratively splits the learned latent spaces, resulting in a tree structure (with leaf nodes at the bottom). In each node, we display the RSA similarity index between 0 (dark blue, not similar at all) and 1 (yellow, identical), where representations are compared in time between the different training stages.

these child nodes start at that stage. When a node is split, the parent node is frozen and learning only continues in leaf nodes (as indicated by the RSA indexes of a parent node being all at 1 after a split). We observe that some child nodes saturate much more quickly than others. For example, node BC 000 saturates only a few training stages after its split from BC 00, while its sibling BC 001 never saturates until the end of the training at stage 50. The RSA analysis of node BC 001 indeed shows that the learned representation continues to evolve as training occurs. This means that this node corresponds to a part of the BC space constituting a rich progress niche for the base module VAE associated with that node. In contrast, node BC 000 will require further splitting to discover such progress niches in its child nodes. The reader can refer to Figure 15 in section F for visualizing discovered patterns in each nodes of the hierarchy.

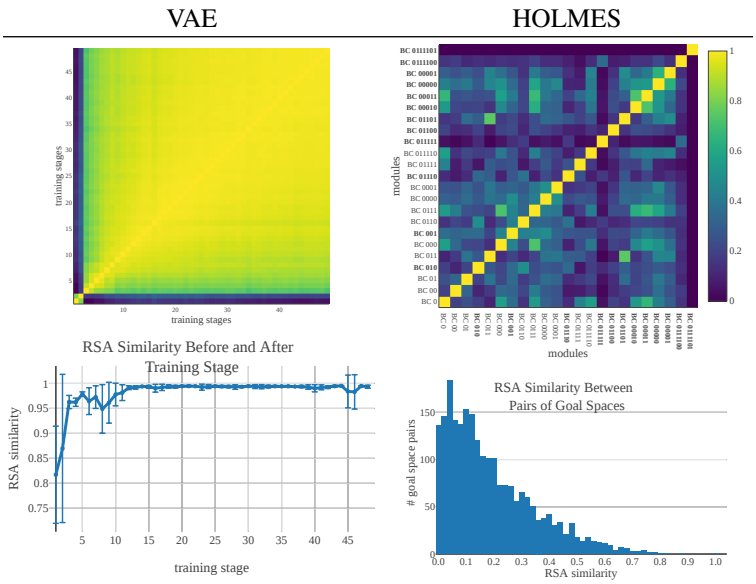


Figure 12: RSA similarity index between 0 (dark blue, not similar at all) and 1 (yellow, identical). (Top) RSA matrix for one experiment as shown in Figure 3 of main paper. (Bottom) statistics over the 10 repetitions: (bottom-VAE) RSA index similarity between representations coming from two consecutive training stages (mean and std); (bottom-HOLMES): histogram of RSA index similarity between all pairs of modules in HOLMES (aggregated over the 10 repetitions).

Figure 12 complements Figure 3 of the main paper with statistical results over 10 repetitions. The statistical results (bottom row) confirm our analysis: the VAE representation saturates quite early in the exploration loop and the representations learned by HOLMES modules are dissimilar from one module to another. Indeed we can see that the VAE representations of all experiments saturate after 15 training stages (high RSA ≈ 1 between remaining consecutive training stages). The histogram of similarity index between all pairs of modules in HOLMES (for all experiments) show a concentration between [0,0.3] (i.e. very low similarity).

D.2 Additional IMGEP baselines with a monolithic BC space

In section 4.2 of the main paper, we compared the incremental training of behavioral training between an IMGEP equipped with a monolithic VAE (IMGEP-VAE) and an IMGEP equipped with the hierarchy of VAEs (IMGEP-HOLMES).

Baselines In this section, we consider different baselines for the training strategy of the monolithic architecture: **BetaVAE** [5], **BetaTCVAE** [9], **TripletCLR** [8, 67], **SimCLR** [11] and **BigVAE**. All the baselines have the same encoder architecture and training procedure than the main baseline IMGEP-VAE (as detailed in section C.3). The baselines differ in their approach to train the encoder network, including several variants of variational-autoencoders and contrastive approaches.

The first two variants BetaVAE [5] and BetaTCVAE [9] build on the VAE framework and augment the VAE objective with the aim to enhance interpretability and disentanglement of the latent variables. Therefore only the training loss of the VAE (see section A.1) differs:

- The BetaVAE objective re-weights the b term by a factor $\beta > 1$:
  $\mathcal{L}_{\text{BETAVAE}}(\theta, \phi; \mathbf{x}, \mathbf{r}) = \underbrace{\mathbb{E}_{\hat{p}(\mathbf{x})}\left(\mathbb{E}_{q_\phi(\mathbf{r}|\mathbf{x})}\left(-\log p_\theta(\mathbf{x}|\mathbf{r})\right)\right)}_{a} + \beta \times \underbrace{\mathbb{E}_{\hat{p}(\mathbf{x})}\left(D_{\text{KL}}\left(q_\phi(\mathbf{r}|\mathbf{x})||p(\mathbf{r})\right)\right)}_{b}$
  Our baseline uses $\beta = 10$.

- The BetaTCVAE objective augments the VAE objective with an additional regularizer that penalizes the *total correlation* (dependencies between the dimensions of the representation):

$$\mathcal{L}_{\text{BETATCVAE}}(\theta, \phi; \mathbf{x}, \mathbf{r}) = \mathbb{E}_{\hat{p}(\mathbf{x})}\left(\mathbb{E}_{q_\phi(\mathbf{r}|\mathbf{x})}\left(-\log p_\theta(\mathbf{x}|\mathbf{r})\right)\right) +$$
$$\alpha \times \underbrace{I_{q_\phi}(\mathbf{x}|\mathbf{r})}_{mutual\ information} + \beta \times \underbrace{TC(q_\phi(\mathbf{r}))}_{total\ correlation} + \gamma \times \underbrace{\sum_j D_{KL}(q_\phi(z_j)||p(z_j))}_{elementwise\ KL}$$

  Because TC is not tractable, they [9] propose two methods based on importance sampling to estimate it: *minibatch weighted sampling* (mws) and *minibatch stratified sampling (mss)*. Our baselines uses $\alpha = 1$, $\beta = 10$, $\gamma = 1$ and mss importance sampling.

The second two variants TripletCLR [8, 67] and SimCLR [11] use contrastive approaches as training strategy for the encoder. Contrary to the VAE variants, these approaches drop the decoder networks and pixel-wise reconstruction as their training objective operates directly in the latent space. The encoders are trained to maximize agreement between differently augmented versions of the same observation o. We used to 2 variants for the contrastive loss:

- Triplet Loss: $\mathcal{L}_{\text{TRIPLETCLR}}(A, P, N) = \max\left(d(R(A), R(P)) - d(R(A), R(N)) + \alpha, 0\right)$ where R is the embedding network, A is an anchor input (pattern o in the training dataset), P is the positive input (augmented version of o), N is the negative input (other pattern o' randomly sampled in the training dataset), $d(\cdot, \cdot)$ is the distance in the latent space (we use cosine similarity) and α is a margin between positive and negative pairs (we use $\alpha = 1$)
- SimCLR Loss: $\mathcal{L}_{\text{SIMCLR}}(A, P) = -\log \frac{\exp sim(z_A, z_P)/\tau}{\sum_N \mathbb{1}_{[N \neq A]} \exp sim(z_A, z_N)/\tau}$ where τ denotes the temperature parameter (we use $\tau = 0.1$); sim the similarity distance (we use cosine similarity); and z represent the latent features onto which operates the contrastive loss. Please note that for this variant the encoder is coupled to a *projection head* network $g(\cdot)$ such that $o \xrightarrow{R} r \xrightarrow{g} z$ (We use g: FC 16→16 + RelU, FC 16→16). Here the positive pair (A, P) is contrasted with all negative pairs (A, N) in the current batch. The final loss is computed across all positive pairs in a mini-batch.

Finally the BigVAE baseline uses the same architecture and training strategy than the main VAE baseline, but with a much larger embedding capacity (368 dims instead of 16 dims) corresponding to the total embedding capacity of HOLMES if we would concatenate its 23 BC latent spaces.

Results The results are summarized in Figure 13 and corroborate with the insights in the main paper.

- Lack of *plasticity* for the all VAE variants, i.e. inability to adapt the learned features to novel niches of patterns. The bias that we observed in the main paper is confirmed in the RSA analysis, even when changing the training objective or the encoding capacity. Interestingly, IMGEP-BetaVAE and IMGEP-BetaTCVAE show the same profile of discovered diversities than VAE (good at finding a diversity of SLPs but bad for TLPs) whereas the IMGEP-BigVAE seems to have a reversed bias (good at finding a diversity of TLPs but bad for SLPs). We attribute this effect to the difficulty of VAEs with low embedding capacity to capture textures with fine-grained structures (i.e. TLPs) whereas when given a higher encoding-capacity they can more accurately represent TLPs. Therefore the variants with small capacity representations seem better suited for exploring diverse SLPs (to the detriment of TLPs) whereas BigVAE seem better suited for exploring diverse TLPs (to the detriment of SLPs).
- Lack of *stability* for all the contrastive variants, where features are drastically different from one training stage to the other. Contrary to the VAE variants, those approaches do not exhibit a strong *bias* in their BC and therefore do not seem to differ much from the *default* diversity found in Lenia (represented with the black curve), at least for the two *types* of diversity measured in Figure 13.

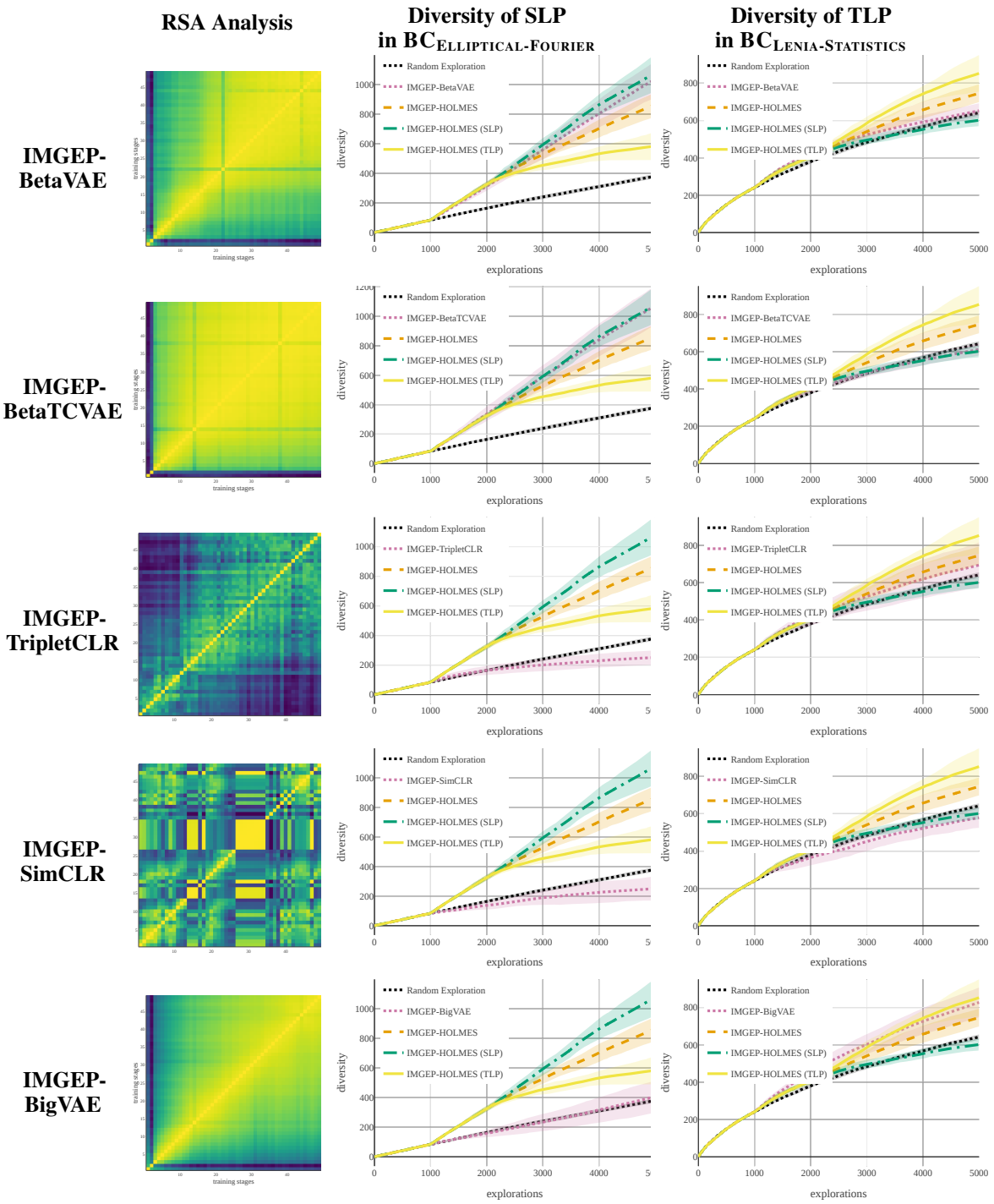


Figure 13: This figure complements the results presented in the main paper, where we replace the baseline with the monolithic BC space (IMGEP-VAE) with different architectures and training strategies. Each row is a baseline denoted as IMGEP-X (where X represents the training strategy used for training the monolithic representation). For each row, we display: (left) RSA matrix where representations are compared in time between the different training stages, as shown in Figure 3 of the main paper; (middle-right) exact same plots than Figure 5 of the main paper where we replace the monolithic IMGEP-VAE baseline (pink curve) by the other baseline (of the current row). Therefore only the pink curve vary between the different graphs of one column.

D.3 Ablation Study: Impact of the Lateral Connections

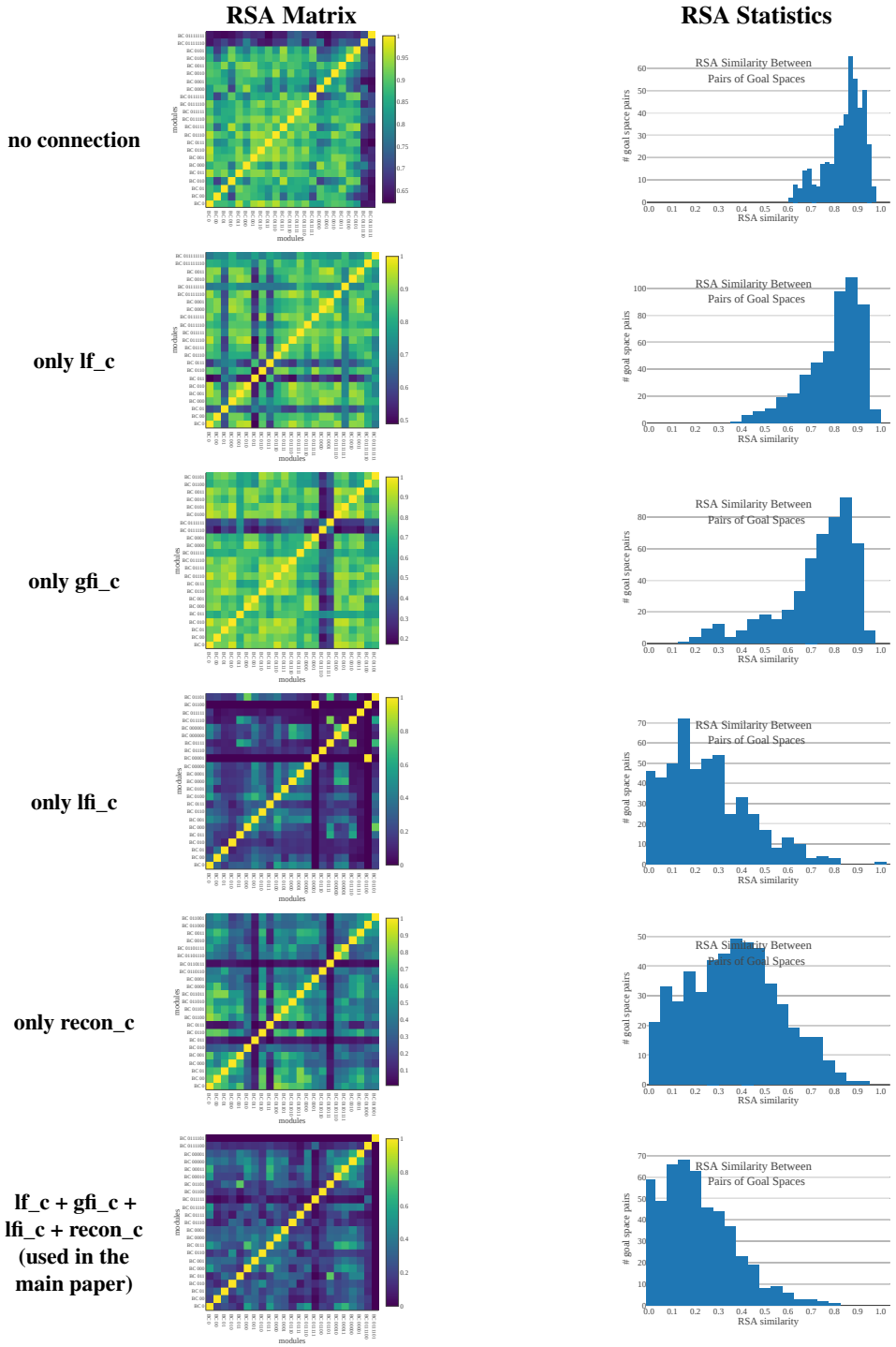


Figure 14: RSA Analysis of the effect of the *lateral connections* on the ability for HOLMES to learn *diverse* module BCs. Each row is an ablation experiment with the corresponding connection scheme. (Left) RSA matrix for one experiment repetition (seed 0), with similarity index between 0 (dark blue, not similar at all) and 1 (yellow, identical). Representations are compared at the end of exploration between the different modules (ordered by their creation time on the left-to-right x-axis). (Right) Histogram of RSA index similarity between all pair of modules (aggregated over the 10 repetitions).

We conducted 5 ablation experiments of the IMGEP-HOLMES variant presented in the main paper, each has 3 repetitions with different seeds. Each ablation experiment considered a different connection scheme with either zero or only one connection among **lf_c**, **gfi_c**, **lfi_c** and **recon_c** (proposed connections in HOLMES, see Figure 6 and Table 5). As shown in Figure 14, the lateral connections are *essential* to learn *diverse* behavioral characterizations among the different modules of the hierarchy. Indeed we can see that IMGEP-HOLMES without any connection (first row in the figure) learns BCs that are highly similar from one module to another (histogram concentrated around [0.8, 1] RSA indexes, i.e. very similar). We can also see that connections toward the last layers of the decoder are seemingly the more important (lfi_c and recon_c) as IMGEP-HOLMES with only one of such connection succeeds to learn dissimilar BCs per module (the histogram is shifted toward lower RSA indexes). However, the connection at the encoder level (lf_c) and close to the embedding level (gfi_c) seem less necessary, or at least alone are not sufficient to allow HOLMES modules escaping the bias inherent to the VAE learning (show a similar histogram of RSA indexes than the no-connection variant). The connection scheme used in the main paper (last row in the figure) seems to be the best suited to learn diverse BCs (histogram concentrated around [0.8, 1] RSA indexes, i.e. very dissimilar).

E Comparison of HOLMES with Related Methods

In this section we provide a comparison of HOLMES with recent work in the literature: CURL [61], CN-FPM [40] and pro-VLAE [45]. Those approaches also propose to dynamically expand the network capacity of a VAE in the context of *continual* representation learning, and therefore share similarities with HOLMES. In table 6, we provide a high-level comparison of the proposed approaches which compare the different architectures according to their *structural bias*, handling of *catastrophic forgetting*, architecture for *dynamic expansion*, handling of *transfer* between the different group of features, criteria for the *expansion trigger*, and if they are performing *data partitioning* (i.e. learn different set of features for different niches of observations).

Table 6: High-level comparison of the general choices of HOLMES with those of previous methods: CURL [61], CN-FPM [40] and pro-VLAE [45]. Please refer to the original papers for more details.

	CURL	**CN-DPM**	**pro-VLAE**	**HOLMES**
Structural Bias	Mixture of Gaussians (in a single VAE latent space)	Dirichlet Process Mixture (flat set of VAE modules)	Hierarchical Levels (in a single VAE)	Hierarchical Mixture (binary tree of VAE modules)
Catastrophic Forgetting	Generative Replay	Freeze	"fade-in" coefficient (same network)	Freeze
Dynamic Expansion	New component in the MoG	New module VAE	New feature layer in the VAE	New module VAE
Transfer	Single Shared Network with several "heads"	Lateral connections (exhaustive as in PNN [64]))	Single Shared Network with several "levels"	Lateral connections (parent-to-children only)
Expansion Trigger	Short-Term Memory Size	Short-Term Memory Size	Predetermined	Node Saturation
Data Partitioning	Soft partitioning	Soft partitioning (coupling of each VAE with discriminator)	None (same network)	Hard Partitionning (boundary in BC_i)

As we can see, while HOLMES shares *conceptual* ideas with those approaches, our approach has key differences:

1. It uses a hierarchy of different latent spaces whereas CURL uses a single latent space, CN-DPM uses a flat set of different latent spaces and pro-VLAE uses a fixed-set of latent spaces (different levels in one network)
2. CURL and CN-DPM show results in the context of continual multi-task classification and demonstrate that their modular architecture can separate well the latents allowing to unsupervisedly discriminate between the different input observations / tasks (eg: discriminate digits in MNIST at test time when they have been sequentially observed at train time). However, CURL does not use different features for the different niches of observations and it is not clear if the flat approach of CN-DPM does learn different features between the different modules. However HOLMES targets to learn dissimilar set of features per BC in order to achieve *meta-diversity*.
3. Pro-VLAE is not applied in the context of continual learning but rather proposes to progressively learn features at different levels in the VAE layers, showing that it can successfully disentangle the features. Even though disentanglement is a key property to avoid redundant features, we believe that it is also key to have diverse set of features for the different niches of observed instances.

F Additional Visualisations

Figure 15 shows examples of patterns discovered by IMGEP-HOLMES (non-guided) within the learned tree hierarchy. The patterns shown in the root node are representative of the diversity of all the discovered patterns in that particular run (100% of the patterns). The boundaries fitted when splitting each non-leaf node (see procedure in section A.1) makes each pattern follow a particular path in the hierarchy, from the root node to a leaf node. Goals are sampled by the IMGEP by first sampling a leaf uniformly among all existing leafs, then sampling uniformly in the hypercube fitted around currently reached goals within that leaf (see section 3.2 of the main paper). However, we observe that the percentage indicated in each node does not reflect this uniformity (for example, only 5.7% of the patterns fall in the leaf BC 001). The interpretation is that leafs with low percentages correspond to unstable niches: when a goal is sampled in such a leaf, the small mutation applied in the parameter-sampling policy is sufficient to produce a pattern which is different enough to fall in another leaf.

We qualitatively observe in Figure 15 that the boundaries fitted during the splitting procedure tend to separate the patterns into visually distinct categories. For example, the proportion of TLPs is much higher in BC 01 compared to BC 00 ; the leaf BC 00000 contains only blank patterns while its sibling BC 00001 contains only SLPs ; the nodes below BC 01111 (bottom-right of the tree) contains only TLPs.

Figures 16 and 17 show discovered patterns when IMGEP-HOLMES is guided towards SLPs or TLPs, respectively, through simulated user feedback as described in section 4.3 of the main paper. We observe that the user guidance is able to dramatically affect both the diversity of the discovered patterns and the structure of the hierarchy . When guided towards SLPs, most of the discovered patterns are SLPs (most TLPs in Figure 16 are concentrated in the leafs BC 01110 and BC 01111 which represent approximately 15% of the discovered patterns). On the contrary, when guided towards TLPs, most of the discovered patterns are TLPs (most SLPs in Figure 17 are concentrated in the leafs BC 000 and BC 001 which represent approximately 34% of the discovered patterns). As a consequence of this bias toward either SLPs or TLPs, we observe that HOLMES has created more branches in the direction of the desired patterns (either SLPs or TLPs) in order to enrich their corresponding representations.

Additional visualisations can be found on the project website `http://mayalenE.github.io/holmes/`.

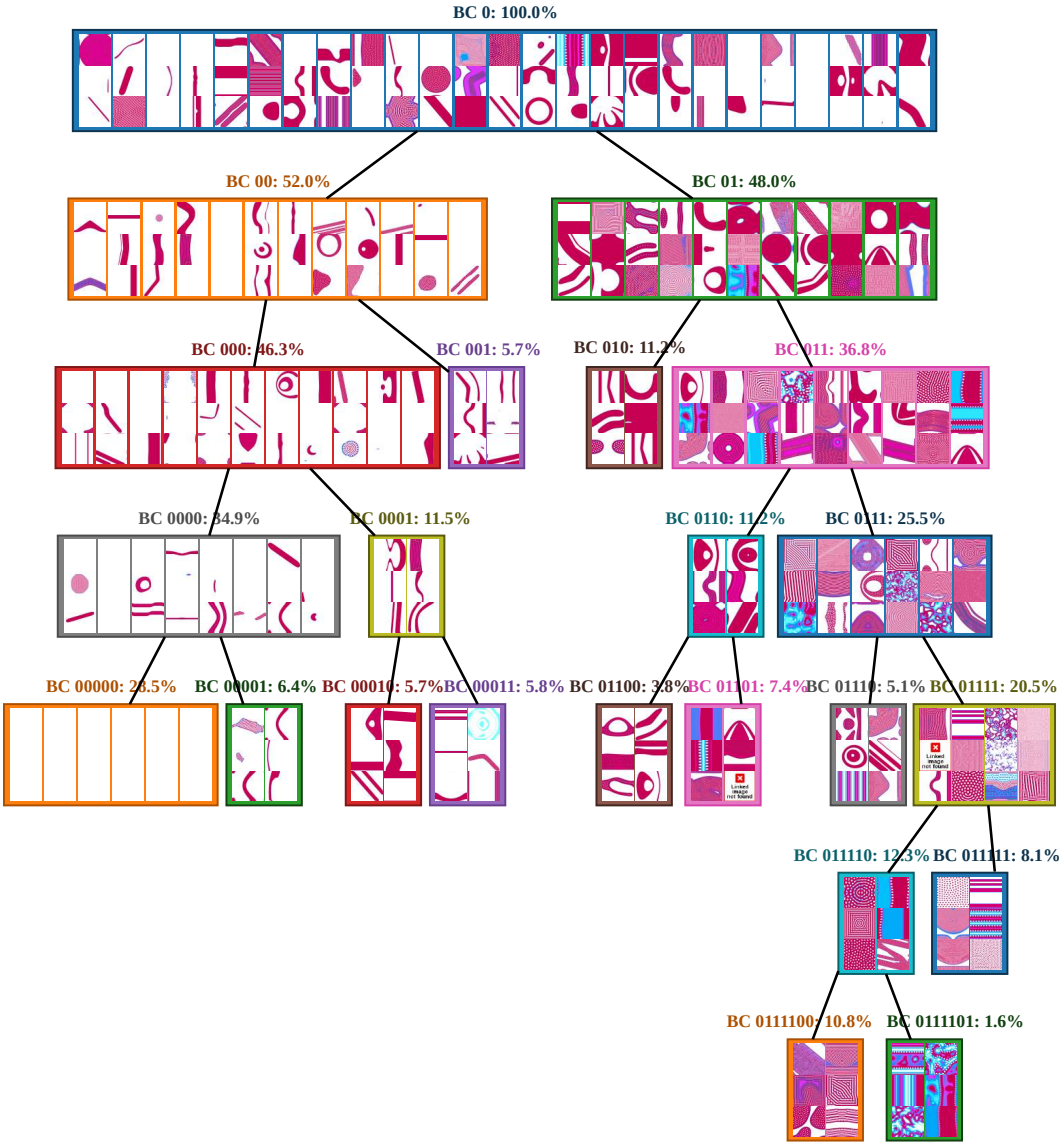


Figure 15: Examples of patterns discovered by IMGEP-HOLMES (non-guided) within the learned tree hierarchy. The hierarchy is the same as in Figure 11. In each node is displayed the percentage of discovered patterns directed through that node, as well as a set of pattern images representative of the diversity within that node (the set is built with the procedure described in section B.3). The number of patterns per node reflects the indicated percentage.

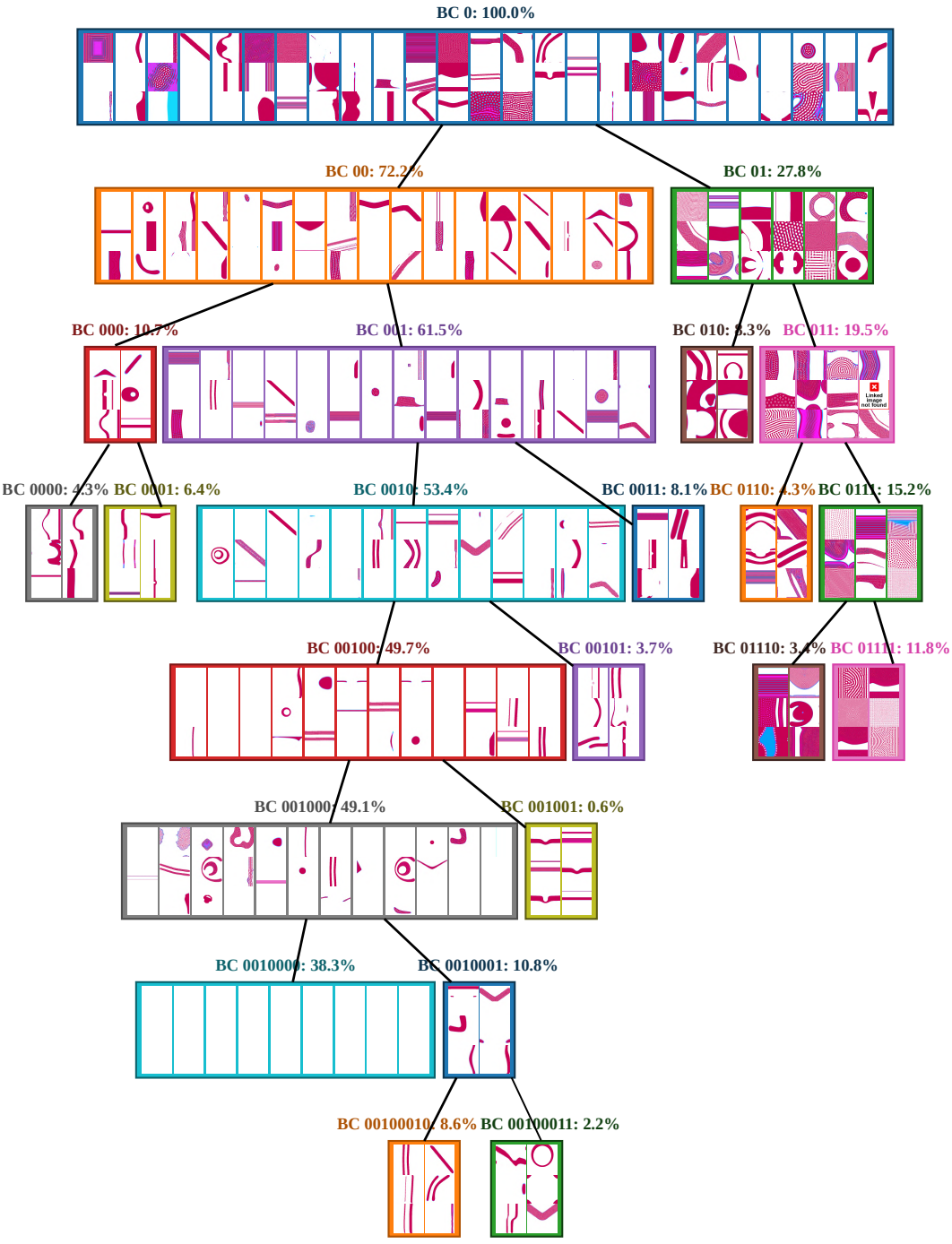


Figure 16: Examples of patterns discovered by IMGEP-HOLMES within the learned tree hierarchy, when guided towards SLPs through simulated user feedback as described in section 4.3 of the main paper. Same convention as in Figure 15.

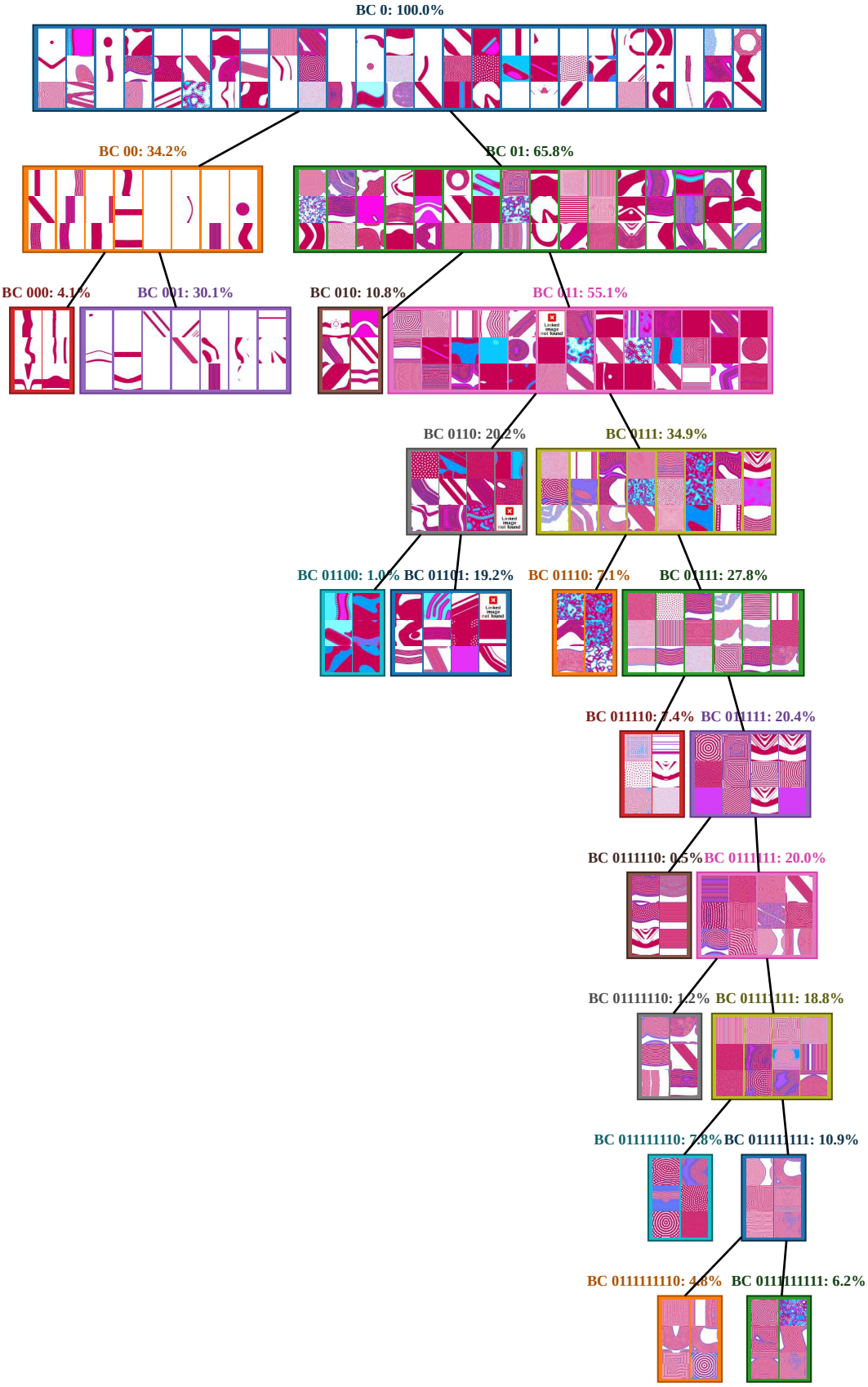


Figure 17: Examples of patterns discovered by IMGEP-HOLMES within the learned tree hierarchy, when guided towards TLPs through simulated user feedback as described in section 4.3 of the main paper. Same convention as in Figure 15.