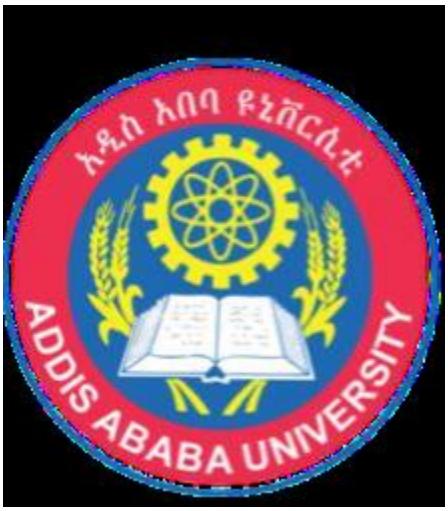

ADDIS ABABA UNIVERSITY
ADDIS ABABA INSTITUTE OF TECHNOLOGY SCHOOL OF ELECTRICAL AND
COMPUTER ENGINEERING

Formulating Goal Spaces from Latent Space Representations of Raw Sensory Inputs for Autonomous Learning of Artificial Agents

By Mekdes Masresha
Advisor Dr. Menore Tekeba

A thesis submitted in partial fulfillment of the requirements for the Master of Science in Computer Engineering

June 2025 Addis Ababa, Ethiopia

ADDIS ABABA UNIVERSITY ADDIS ABABA INSTITUTE OF TECHNOLOGY SCHOOL OF ELECTRICAL AND COMPUTER ENGINEERING
SCHOOL OF ELECTRICAL AND COMPUTER ENGINEERING

This certifies the validity of Mekdes Masresha's thesis, "Forming Goal Spaces from Latent Space Representations of Raw Sensory Inputs for Autonomous Learning of Artificial Agents" which was submitted to partially fulfill the requirements for the Master of Science degree, complied with university regulations and met recognized standards for quality and originality.

Approved By the Board of Examiners and Advisors

Dean, (SECE), AAiT	Signature	Date
Advisor Menore Tekeba (Ph. D)	Signature	Date
Thesis Examiner	Signature	Date
Thesis Examiner	Signature	Date

Acknowledgments

I am very thankful to my advisor, Dr. Menore Tekeba, for his constant support, helpful guidance, and valuable feedback during this research. His knowledge and Encouragement have been essential in developing this work, and I truly appreciate the chance to learn from him.

Declaration of Authorship

I attest that the study "Formulating Goal Spaces from Latent Space Representations of Raw Sensory Inputs for Autonomous Learning of Artificial Agents" is entirely original with my own writing. The work hasn't been submitted for evaluation elsewhere. Any instances in which content has been borrowed from another source have been appropriately cited and acknowledged.

Researcher's Name	Signature	Date
Mekdes Masresha	_____	_____

Contents

Abstract

This paper presents a method for enabling autonomous agents to specify their own goals from structure in raw sensory data. Instead of relying on prespecified goal and behavior, agents use intrinsic reward mechanisms to find and pursue goals in complex, dynamic worlds where explicit goal specification is impossible. We emphasize the role of latent space as a meaningful and structured representation of perceptual input. We use Variational Autoencoders (VAEs) to transform raw sensory input into structured latent representations, which serve as the foundation for autonomous goal generation. Self-generated goals allow for exploration, learning, and adaptability in the absence of explicit supervision.

Our model incorporates unsupervised learning for latent space creation with Reinforcement Learning (RL), employing Proximal Policy Optimization (PPO) for policy learning. Our approach efficiently to diverse environments through the balance of exploration and exploitation. Experimental results demonstrate the effectiveness of our approach. On the CartPole Swing-Up task, our approach achieved 874.84 with a 100% success rate. For the Cheetah Run task, it achieved 734.22 and a 98.95% success rate. For the Hopper task, it demonstrated good sample efficiency and an 89.67% success rate, although it was somewhat slower to converge than some of the other approaches.

This work takes a step forward in the development of autonomous systems that are both scalable and flexible. We show that latent space representations combined with intrinsic motivation offer a great way to go in building agents capable of goal-directed behavior in dynamic and unstructured environments.

Keywords: goal space, latent space, autonomous learning, reinforcement learning, intrinsic motivation

List of Figures

List of Tables

List of Abbreviations

Abbreviation	Full Form
GSF(ExO)	Goal Space Formulation from External Observation (our method)
RL	Reinforcement Learning
PPO	Proximal Policy Optimization
TRPO	Trust Region Policy Optimization
AI	Artificial Intelligence
ReLU	Rectified Linear Unit
IR	Intrinsic Reward
VAE	Variational Autoencoder
MDP	Markov Decision Process
DMRL	DeepMind Reinforcement Learning
DM	DeepMind

CHAPTER 1

Introduction

This paper aims to discuss how we can allow artificial agents to set and achieve goals autonomously by utilizing latent space representations of raw sensory input, via intrinsic reward approaches [7]. One of the challenges of reinforcement learning and autonomous agent design is the reliance on external rewards, which are typically manually designed and specific too individual tasks [12].

This dependence on extrinsic rewards limits scalability and flexibility, since it takes human effort to define objectives and shape reward functions for novel tasks or environments [11]. To address this, we present a framework that replaces extrinsic task-specific rewards with intrinsic rewards, which allows agents to explore, discover, and learn independently [2]. To illustrate this concept, imagine we place a baby in a room with no instructions, no reward, and no goals; something incredible still happens: the baby explores. It plays with its hands, tries to stand up, babbles randomly, and soon learns to walk, speak, and navigate its world. Nobody taught it to do those very things. No scoreboard when it takes its first steps. The inspiration originates within oneself, curiosity, surprise, and the pleasure of learning in itself. Now, imagine we would like an AI agent to behave similarly. Imagine placing it in a rule less, command less, goal less world, just a stream of sensory inputs. Like the baby, the agent starts off knowing nothing about its task, environment, or actions available.

For example, imagine a simulated cheetah robot that is put into an environment with no instructions, no pre-defined objectives, or extrinsic rewards. It does not know what "running fast" means or how to do it. Instead, it must decipher its raw sensory inputs and figure out what behaviors lead to meaningful progress on its own. Could it learn to do something useful on its own? Could it learn to walk, stabilize, roam, and even solve problems, without us telling it what to do? This is the central question this thesis is trying to answer.

Without explicit external rewards, the agent must rely on intrinsic motivations to explore its environment and autonomously find rewarding behaviors [11]. Intrinsic rewards are internal drives, that motivate the agent to try actions, discover patterns in its sensory input, and hone its skills over time [2]. The agents begin by making random movements, yet it is eventually possible for it to move towards stability and even learn to stand or walk, driven by nothing more than intrinsic rewards that track progress and novelty [10].

Latent space representations are central to such self-learning [7]. By condensing raw sensory inputs to a more tractable lower-dimensional latent space by techniques, Variational Autoencoders, (VAEs) [6] or self-supervised learning (Pathak et al., 2017) [2], the agent can learn to find the underlying structure of its sensory data. These latent representations provide the agent with a useful abstraction that enables it to identify areas of interest or possible goals in its environment [11]. The real challenge, however, is to find actionable and goal spaces in the latent space, especially without external supervision [10].

Our approach integrates latent space representations and intrinsic reward mechanisms to surmount these challenges [3]. We employ intrinsic rewards such as goal discovery (encouraging the agent to find new goals or states) and learning progress (rewarding quantitative improvements in skill discovery) [2]. These rewards enable the agent to explore latent spaces freely, discover emergent goal spaces, and iteratively build its behaviors [10]. Intrinsic rewards also enable the agent to remain motivated to explore its world, learn new things, and build its skills, even in the absence of any external goals or rewards defined [11].

The main contribution of this paper is the development of a methodology that combines intrinsic reward systems with latent space representations for autonomous goal discovery. We show how intrinsic rewards, including goal discovery and learning progress, can be utilized to explore and map latent spaces, allowing the agent to find and master useful goals. We also provide a framework for quantifying the agent's learning progress to keep it focused on acquiring skills and avoid stagnation or redundant exploration.

In the following sections, we cover the theoretical foundations of intrinsic rewards and latent space representations, describe how goal spaces are specified, and outline the implementation of our approach. With experimental validation, we show how this method enables agents to learn and execute complex behaviors like standing and walking autonomously, without prior knowledge or external supervision. By combining intrinsic motivation with latent space learning, this work advances the field of autonomous learning, offering a solid framework for agents to achieve scalable, task-agnostic self-learning across diverse environments.

It is one of the grand challenges in artificial intelligence and reinforcement learning to enable artificial agents to automatically learn and carry out tasks or goals without explicit supervision [11]. The standard learning methods typically rely on reward signals that are carefully designed for a task and externally given [12]. These reward signals guide the agent towards certain outcomes, i.e., standing, walking, or striking a target, and require human effort for task specification and the creation of appropriate reward functions. Although such approaches work well under controlled settings where goals are well-specified, they fail to scale and generalize to open-ended or dynamic environments where goals are unknown [10].

To allow agents to learn autonomously, they must be able to explore their world, discover worthwhile goals, and learn the skills necessary to achieve those goals without any external supervision [11]. However, such autonomy is challenging, particularly when agents have no prior knowledge about their world, actions, or tasks [10]. For instance, an agent that is unaware of standing or walking must experiment with its actions to learn stable stances and determine how to walk (Pathak et al., 2017) [2]. The lack of explicit task specifications and extrinsic rewards makes it increasingly challenging for the agent to know where to explore or how to reward progress towards beneficial goals [12].

Another key challenge is to use latent space representations of raw sensory inputs to learn goals. Latent spaces, learned via approaches like Variational Autoencoders (VAEs) [6], provide a compressed and structured representation of complex sensory data. These representations capture the inherent structure of inputs and help to identify potential regions of interest [7]. However, existing methods fail to link such latent representations to actual, actionable goals. While latent

spaces are great at abstracting out complex data, they do not necessarily reveal which regions of the space map to useful tasks or behaviors [12].

Most current methods that use latent spaces for reinforcement learning still rely on prespecified goals or external labels to guide exploration [7]. These methods do not fully leverage the potential of latent spaces for autonomous goal discovery. Additionally, they lack mechanisms for dynamically prioritizing new and promising regions in the latent space, making exploration and skill discovery less efficient [12]. As a result, artificial agents fail to autonomously learn and act in a flexible and scalable manner.

This mismatch calls for an approach that can bridge the divide between intrinsic motivation and learned latent representations even when all the agent has to work with are raw observations, i.e., camera images. In our method, the agent learns a representation of its world by acquiring a dense, internal world representation directly from observations. It then uses intrinsic rewards, including curiosity and learning progress, to guide it, allowing it to explore, learn, and stay motivated without any external supervision [3, 10]. By associating what the agent sees with what it finds interesting or new, we allow it to discover its own goals and keep getting better over time. This Designing Goal Spaces from Latent Space Representations of Raw Sensory Inputs for Autonomous Learning of Artificial Agents4makes the approach strongly applicable to real-world, messy situations where there are no pre-specified tasks or hand-engineered rewards [11].

Despite the promise of learning from latent spaces, most existing approaches fail to effectively integrate intrinsic rewards for bridging abstract latent representations and actionable goal discovery [3]. Most current approaches still rely on pre-defined goals or extrinsic guidance to navigate latent spaces, limiting the agent's ability for autonomous discovery and prioritization of new or reachable areas [2]. Furthermore, these kinds of methods do not adapt exploration dynamically based on the agent's learning progress, leading to inefficiency in skill learning and adaptation to the environment [10].

This work aims to address these limitations by incorporating intrinsic reward mechanisms that balancing learning performance (measuring improvements in skill acquisition) and goal

discovery (identifying and exploring new or significant areas in the latent space) [2]. By rewarding the agent for advancements in its understanding of the world and for achieving new goals, intrinsic rewards can more effectively guide exploration and recursively improve the agent's behaviors [3]. For example, an agent that starts from scratch with no knowledge about standing and walking can use intrinsic rewards to experiment with different postures and movements, discover stable configurations, and learn to walk without any external supervision. At its simplest level, this challenge entails enabling artificial agents to find and achieve tasks or goals by themselves without explicit supervision, a problem that has been further compounded by the lack of effective use of latent spaces in goal discovery [7]. Utilization of intrinsic rewards provides a nice solution to address this issue [10].

1.2 Objective

1.2.1 General Objective

To develop, investigate, and experiment on the goal space design from the dense latent space representation of sensorimotor percept data and enable autonomous learning of an agent via the intrinsic reward model.

1.2.2 Specific Objective

- Selecting the agent and its environment
- Selecting the most relevant input of the goal space formulator network
- To identify the most informative feature dimension of the latent space representation for goal space definition
- To identify an intrinsic reward model
- Implement the agent, the RL algorithm(s), the representation learning network, and the intrinsic reward model.
- Train all the modules of the agent with necessary training parameters
- Test, evaluate, and discuss the performance of the self-learning agent.

1.3 Scope

The thesis aims to formulate, explore, and examine a framework that facilitates independent learning by leveraging dense latent space representations of sensory data. The research will entail formulating a goal space design with the aid of these latent representations and exploring how intrinsic reward models can be utilized in motivating exploration and learning in agents.

Experimental validation will be done in simulated environments to evaluate the effectiveness of the proposed approach in goal-directed behavior and learning efficiency. While the thesis will not cover real hardware testing, advanced multimodal sensor fusion, or long-term deployment, it will emphasize proof-of-concept experiments and theoretical contributions at the frontier of representation learning and reinforcement learning.

1.4 Research Methodology

This project is founded on three basic conceptual pillars:

Representation Learning: Such models compress raw sensory inputs (joint angles, camera images) into compact latent vectors. These vectors form the agent's internal representation of the world and are the space from which goals are sampled.

Intrinsic Motivation Theory: Intrinsic motivation theory Inspired from cognitive science, this theory proposes that agents (artificial or biological) may learn by optimizing internal drives such as curiosity, novelty, or competence. These drives are computationally formalized as reward signals independent of external goals.

Goal-Conditioned Reinforcement Learning (GCRL): This formalism allows agents to condition actions on representations of goals such that it is possible to learn a single policy that generalizes too many goals, including goals defined in latent space.

Why this Framework?

Alternative methods either have no goal-directed structure or require human involvement. The proposed framework is preferred because:

- It enables unsupervised learning of useful goals.
- It combines task-agnostic exploration with goal-directed learning.
- It can be generalized to new environments without re-defining objectives.

The research methodology of this thesis is one that takes a direct, step-by-step approach in an attempt to achieve the stipulated objectives. It is categorized into some principal stages:

a) **Literature Review:** The first step involves conducting a critical literature review to determine the state of the art in research into self-directed learning, reinforcement learning (RL), goal space formulation, latent space representation, and intrinsic reward models. The review will highlight gaps in the existing approaches and guide the construction of the proposed method in such a way that it builds on what has previously existed and adds new, original contributions [7][12].

b) **Proposed Approach:** A new approach will be developed based on the findings of the literature review. This will involve selecting a suitable agent and environment, building the goal space, and implementing an intrinsic reward model. The objective is to build an agent that can learn autonomously using compact latent space representations of perception. The approach will be focused on selecting the most appropriate sensory inputs for the goal space formulation network and picking the best dimensions of the latent space for best goal space formation [10][2].

c) **Data Preparation and Collection:** Relevant sensory data (visual, audio, or other sensor data) will be collected and generated in an artificial environment. The data will be preprocessed and restructured in suitable forms to train the agent. It is desired to have clean, relevant, and high-quality data to facilitate the learning process [11].

d) **Deployment of the Proposed Method:** The agent, reinforcement learning techniques, latent space representation models (like VAE or generative models), and intrinsic reward model will be deployed in this step. The agent will be interacting with the chosen environment, learning by exploration, and learning how to build goals out of the sensory data and intrinsic rewards. The best appropriate RL algorithm will be selected based on how well it fits the task and ability to utilize the intrinsic reward model to direct the learning of the agent [7][12].

e) **Agent Training:** The agent will be exposed to a training session as soon as components are ready. Latent space representation network, RL algorithm, and intrinsic reward model will be trained using appropriately chosen parameters during this process. Hyperparameters will be tuned to optimize performance, and the agent will undergo several learning cycles to improve its goal setting abilities and autonomous goal achievement [10][12].

f) **Testing, Evaluation, and Results Discussion:** After training, the agent will be tested in the chosen environment to assess its ability to autonomously learn and achieve goals. The criteria of

evaluation will be goal achievement efficacy, learning rate, and the capability of the intrinsic reward model to encourage exploration of the agent. The results will be contrasted with existing methods and evaluated for strengths, weaknesses, and where future changes and research can be explored [11][3]. By following this methodology, the research aims to come up with a solid framework for autonomous learning using latent space representations, goal space construction, and intrinsic rewards, with extensive testing and evaluation of framework performance.

1.5 Contribution

The primary contribution of this paper is listed below:

- We apply a method for agents to figure out their own capabilities and goals simply by watching and trying things out without our having to hand-program specific tasks or rewards.
- As opposed to being founded on extrinsic rewards, our method relies on intrinsic motivation, agents are incentivized to get closer and discover new things.
- Through transforming raw sensory data into a lower-dimensional, easier-to-understand form (a latent space), agents can better understand their environment and establish their own goals.
- This allows them to actively pursue optimizing behavior in the moment, focusing on what's most useful or new to them in their environment.
- Our approach allows agents to learn from scratch-with no prior knowledge-and then learn complex actions like standing or walking.
- We've built a scalable and flexible system that can handle changing environments and new challenges without needing constant human input.

Overall, this work brings us closer to creating autonomous learning systems, with big potential for AI, robotics, and beyond.

1.6 Thesis Organization

This thesis is organized into six chapters. Chapter 1 describes the research problem, objectives, and rationale for research, emphasizing the challenges and importance of goal space formulation from latent space representations for autonomous learning. Chapter 2 introduces the background

theory, such as key concepts like latent spaces, goal space formulation, and reinforcement learning, and the relevant methodologies like intrinsic motivation and unsupervised learning. Chapter 3 discusses related work, emphasizing state-of-the-art techniques, their constraints, and gaps to which this study aims to contribute. Chapter 4 narrates the intended approach, specifying raw sensory data processing methodology, goal space building, and policy optimization through reinforcement learning and intrinsic rewards. Chapter 5 dives into the experimental results, highlighting how our framework performs across different benchmarks in terms of efficiency, adaptability, and scalability. It paints a clear picture of where our approach stands compared to existing methods. Finally, Chapter 6 brings everything together by reflecting on the main takeaways, highlighting the broader impact of the work, and pointing toward exciting future directions to push autonomous learning systems even further.

CHAPTER 2

Background

The present chapter locates the key concepts and theoretical foundations needed to understand the research on learning goal spaces from latent space representations of raw sensory inputs to autonomous learning in artificial agents. Autonomous learning for autonomous agents requires an integrated view of how sensory input, internal representation, and motivational signals are integrated to produce intelligent behavior. This chapter outlines the theoretical foundation of our solution by overviewing the key components and demonstrating how they integrate into our solution.

2.1 Central Concepts and Their Integration

At the heart of this thesis is a framework that allows agents to learn from raw sensory input by projecting these observations onto a latent space, learning goals from these representations, and learning how to achieve them from intrinsic rewards. It all occurs within an RL system based on PPO, in the formalism of an MDP.
Below is a top-level explanation of how these pieces fit together:

Conceptual Flow:
1. Raw Sensory Data (i.e., vision, proprioception)
2. Latent Representation with VAE
3. Goal Space Discovery from latent clusters
4. Intrinsic Rewards to encourage exploration and learning.
5. Policy Optimization with PPO in MDP framework

2.2 Goal Spaces and Autonomous Learning

Autonomous learning is important for artificial agents acting in complex environments, particularly where little or no external rewards are present. Intrinsic motivation has also proven a strong paradigm to support effective exploration and learning in such agents [2]. By using

intrinsic reward techniques, agents can define their own goals and track their performance according to internal measurements [3]. A critical part of doing this is creating goal space representations, which identify the preferred states the agent seeks to achieve [7]. A helpful strategy for establishing these goal spaces is by means of using latent space representations of raw sensor information [6].

Goal Spaces from Latent Space Representations of Unprocessed Sensory Inputs for Autonomous Learning of Artificial Agents Attaching autonomous learning with goal space discovery is of enormous significance for AI systems. It enables agents to act autonomously in open-ended environments where they are not restricted by pre-specified tasks [7]. In these environments, the agents can learn new tasks automatically, set goals, and strive towards them through trial-and-error learning. The outcome is more adaptive and scalable systems as the agent is learning and adapting constantly to new challenges [6].

One of the greatest advancements in this field is intrinsic motivation, where not only are the agents rewarded when they achieve certain goals but also for finding new goals or acquiring new skills in order to encourage long-term growth [3]. This method contrasts with traditional reward schemes that address only the achievement of specified goals, hence limiting the agent's learning potential to those goals [2]. As noted by Oudeyer et al. (2007) [11], intrinsic motivation systems entice agents to be exploratory, self-developing, and curiosity-driven, rather than merely achieve task based rewards. This method entices agents to try new states, expanding their goal spaces and increasing their dynamic adaptability and resilience under dynamic circumstances [11].

2.3 Latent Space and Variational Autoencoders (VAEs)

Latent space representations are abstracted, low-dimensional representations of raw sensory data, such as images or proprioception data, projected to lower-dimensional spaces but preserving the essential properties of the environment. In reinforcement learning, latent spaces are compressed representations of environment states, actions, or observations. Rather than processing raw inputs like game pixels or robot sensor readings directly, the agent works within a latent space, through which it can learn more meaningful patterns. This reduces computational complexity and enables

the agent to generalize more across tasks or environments [7].The latent space representation theory is core to the World Models approach, by Ha and Schmidhuber (2018) [6]. World Models enable agents to construct a small, generative model of their world, allowing them to simulate and forecast future states without needing real time interaction. Models are usually constructed in a latent space, supporting more effective exploration and planning. To create these latent representations, World Models utilize Variational Autoencoders (VAEs), which enhance. The planning and decision-making abilities of the agent in challenging worlds to simulate [6].

Variational Autoencoder (VAE) is a cutting-edge model that can learn latent space representations of high-dimensional data. As a generative model, it maps challenging data distributions such as those found in images, text, or audio into a continuous latent space [6]. This transformation allows VAEs to learn in a lower-dimensional space, which is applicable to tasks like data generation, compression, and feature learning [6].

The structure of a VAE is defined by two elements: the decoder and the encoder. The encoder transforms high-dimensional input information (images or sequences) to the latent space representation, while the decoder transforms from the latent representation to the data. One of the defining aspects of VAEs is the particular manner in which the latent space itself is modeled [6].

2.3.1 Applications and Importance of Latent Space Representations

In the reinforcement learning, the latent space representation is of the main advantages for improving exploration and generalization. Agents using such representations can better navigate new cases by understanding common patterns across all states or environments. For instance, agents can convert raw sensory signals (images) to a latent space (latent symbol), which enables agents to recognize and achieve their goals. These approaches include deep reinforcement learning (DRL) and unsupervised learning and are frequently utilized to autonomously learn these latent space representations. This includes approaches like autoencoders and variational autoencoders (VAEs) that produce compressed and lower-dimensional representations of high-dimensional data, making it easier for agents to determine which features of their environment to focus on when making decisions. The obtained latent space is subsequently a basis to construct

goal spaces and offers a computationally efficient approach to encode possible goals structures and the transitions between them [1][2].

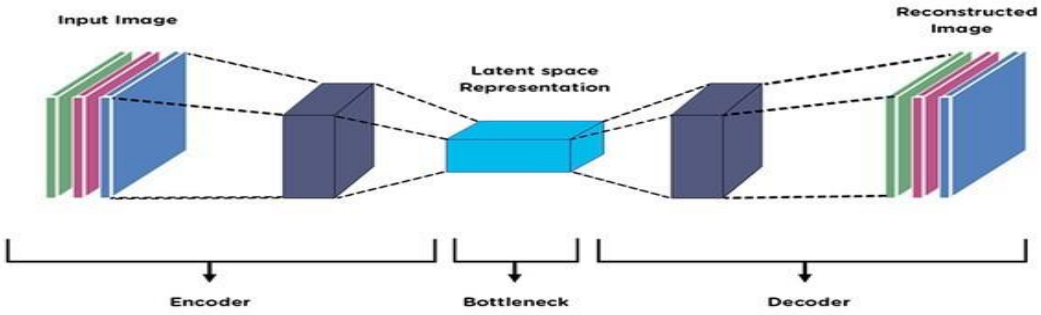

Figure 2.1: Latent Space Representations

2.4 Intrinsic Reward Techniques

Intrinsic rewards are internal cues that guide agent behavior in the absence of external feedback. They promote exploration and self-directed learning. Common mechanisms include:

• **Curiosity-Driven Exploration**: Prediction error rewards, wherein agents are motivated to explore areas of uncertainty within the environment [2][12][25].

• **Novelty-Seeking**: Points for traveling to previously undiscovered states [21][25]. Learning Progress: Rewards for improvement or reduction in uncertainty [15][40].

• **Empowerment:** Incentives to gain greater control of the environment [14][15].

In the goal space design, intrinsic rewards allow agents to discover different areas of the latent space, and ultimately, to discover and design their own goals [1][13][20][40].

2.4.1 Intrinsic Rewards vs. External Rewards in Goal Space Formulation

Intrinsic rewards bring with them unique benefits and challenges, particularly when used for the intention of formulating goal spaces from latent space representations within autonomous learning.

While they present challenges, their capacity to drive autonomous exploration, goal discovery, and learning effective latent representations puts them at the vanguard of guiding goal spaces. Through overcoming these challenges, intrinsic rewards can create highly adaptable, self-

motivated artificial agents capable of learning and performing well in intricate environments [3][15][20].

Aspect	Intrinsic Rewards	External Rewards
Nature	Task-agnostic, promotes exploration	Task-specific, promotes exploitation
Goal Discovery	Enables autonomous goal formulation	Relies on predefined goals Reward
Availability	Always present during exploration	Sparse or delayed in many tasks
Generalization	High potential for transfer learning	Limited to specific tasks or environments
Design Complexity	Challenging to define meaningful metrics	Requires domain expertise for task design
Scalability	Scalable to unknown and dynamic tasks	Struggles in multi-task or dynamic setups
Learning Stability	Can cause instability with poor metrics	More stable with well-defined rewards

Table 2.1: Intrinsic Rewards vs. External Rewards

2.5 Markov Decision Process in Latent Space

A Markov Decision Process (MDP) is a basic paradigm for decision-making under uncertainty. Applied to latent spaces, MDPs abstract away high-dimensional raw inputs at orders of magnitude reduced computational cost [12]. Abstracting away helps in independent learning, allowing agents to act based on compact yet informative representations of the system. In a standard MDP, an agent interacts with an environment and makes decisions in an attempt to maximize some reward signal over a sequence of time steps.

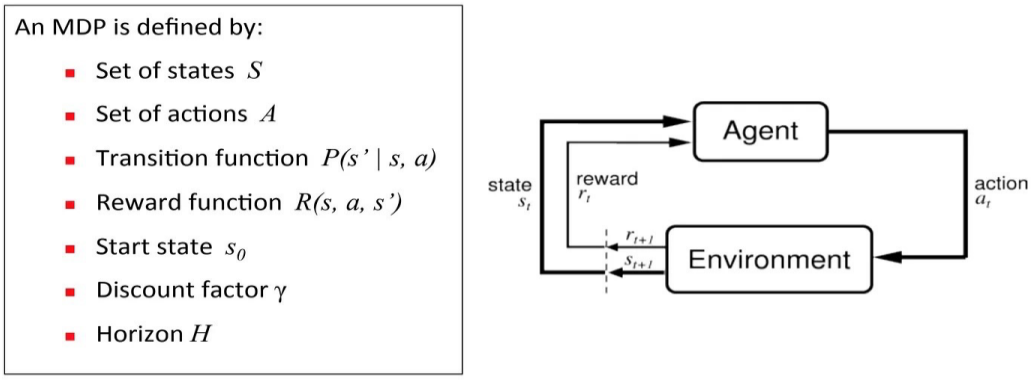

Figure 2.2 Reinforcement Learning Cycle [38]

In latent space, MDP defines as:

States normally represent observable environment configurations. However, in many cases, the environment can be high-dimensional and complex, and the agent may not have access to the complete observable state space. Instead, the agent may act in a latent space, in which the system dynamics and state representations are discovered or inferred [1][2].

a) **Latent Space Representation:** Instead of raw high-dimensional data, the environment is encoded in a lower-dimensional latent space. The representation is usually learned using tools like Variational Autoencoders (VAEs), which attempt to construct a probabilistic model of the environment.

b) **States in the Latent Space:** The states, which are denoted as S, are represented as points in this latent space. The states are typically generated by an encoder that converts raw observations into more abstract features. This allows the agent to operate in a more convenient and compact space than the original high-dimensional state space. The latent space captures the underlying features and dynamics of the environment which enable the agent to generalize, compute, and explore more effectively [5][11][17][35].

c) **Transition and Reward Functions:** The transition function, T(s, a), which describe show the environment changes after the agent takes an action, can be learned within the latent space. The reward function, R(s, a), can also be defined in terms of latent space representations. This can potentially simplify the reward signal, enabling the agent to focus on more abstract goals [13][20].

d) **Policy Learning:** The agent's policy, $\pi(as)$, which dictates the actions to take based on the state, is learned within the latent space. The policy is normally trained using reinforcement learning algorithms like Q-learning, Policy Gradient Methods, or Actor Critic Methods, yet the agent operates over the latent space [6][7][24][31]. Training the policy in this condensed space, the agent can learn more robust policies that generalize better to new, unseen situations.

2.6 Proximal Policy Optimization

Proximal Policy Optimization (PPO) is a well-known reinforcement learning (RL) algorithm, valued for its simplicity, effectiveness, and strong performance in many various fields, including robotics, game environments, and simulated control problems. PPO belongs to the policy gradient class of algorithms, where the aim is to optimize a parameterized policy to maximize cumulative rewards. One of its advantages is how it handles exploration and exploitation balance.

It does this by using a clipped objective function, which limits significant policy updates and prevents disruptive changes that have the potential to destabilize learning. PPO is particularly effective in domains with large or continuous action and state spaces, such as robotics or control tasks like the DeepMind Control Suite [12]. Furthermore, PPO's excellent performance and ease of use have established it as researcher-go-to and real-world go-to alternative in reinforcement learning [12]. Here, PPO is combined with latent space learning and intrinsic rewards to enable goal-directed autonomous learning. By employing PPO in latent spaces, the method supports agents efficiently discover and reach significant goals in complex scenarios, as demonstrated in problems like Hopper, Cartpole Swingup, and Cheetah Run of the DM Control Suite [12].

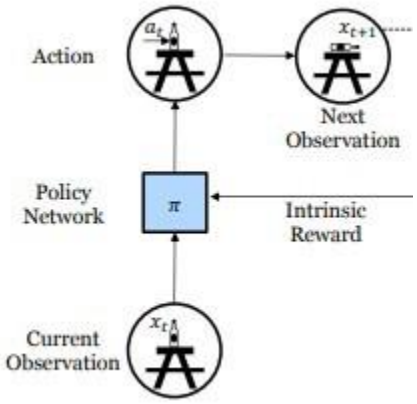

Figure 2.3: Action Selection Process

Advantages of PPO

We use PPO (Proximal Policy Optimization) as the backbone of the intrinsic reward policy learner in our approach, due to several advantages it possesses over other model-free reinforcement learning algorithms:

- Ease of Implementation: It is simple to implement and requires less tuning compared to more complex algorithms like TRPO and hence can be utilized in a wide class of applications.
- Stability: The PPO clipping method stabilizes training by preventing enormous, destabilizing policy updates.
- Efficiency: PPO avoids computationally expensive constraints of TRPO, but achieves similar or even superior performance.
- Scalability: PPO performs well in large-scale settings, including continuous action space scenarios.

Space	Action	Observation
Discrete	√	√
Box	√	√
MultiDiscrete	√	√
MultiBinary	√	√
Dict	**X**	√

Table 2.2: Proximal Policy Optimization PPO

PPO is a significant breakthrough in reinforcement learning that yields a new combination of simplicity, scalability, and stability. It has fast gained attention as an elite algorithm for research and practical applications, achieving drastic improvements in robotics, game AI, and systems of intricate decision-making.

Together, these concepts form a tightly coupled system:

- VAEs enable abstraction of latent states.
- Goal discovery is based on clustering in latent space.
- Intrinsic motivation drives learning and exploration without task-specific supervision.
- MDP structure provides decision-making formalism.
- PPO efficiently optimizes agent behavior in this context.

This foundation allows for a scalable and adaptive autonomous agent to learn in diverse, unstructured worlds.

CHAPTER 3

Related Works

This chapter overviews significant autonomous learning methods, with emphasis given to those employing latent representations, intrinsic motivation, and model-based reinforcement learning. The chapter is also critical of their strengths, limitations, and relevance to the research agenda.

This chapter concludes by placing GSF(ExO) as a necessary contribution to the literature.
Here, we discuss seminal works and recent developments in reinforcement learning (RL) that guide the construction of goal spaces from latent space representations. It emphasizes intrinsic motivation-based methods, latent dynamics models, and unsupervised exploration to facilitate autonomous learning. We present these studies in terms of methodology, contribution, and limitation and highlight the gap filled by this work. Major papers include Intrinsic Motivation in Reinforcement Learning [11], Dream to Control: Learning Behaviors by Latent Imagination [3], Learning Latent Dynamics for Planning from Pixels [7], and Plan2Explore in Unsupervised World Models [2].

They make significant contributions to how latent space representations and intrinsic motivation can be utilized to guide goal formulation and exploration in reinforcement learning. These emphasize the value of self-supervised learning, latent space dynamics, and intrinsic rewards in enabling agents to learn to discover and chase goals autonomously. Most notable is the use of intrinsic motivation techniques to enhance exploration through rewarding agents for new experiences [11], and latent space models like Dream to Control [3] and Learning Latent Dynamics [7] providing solid frameworks for planning and goal discovery. The Plan2Explore framework [2] further exemplifies this by showing how goal creation and achievement can be facilitated through unsupervised exploration, demonstrating the ability of RL agents to find and pursue goals without direct supervision or pre-specified rewards.

3.1 Intrinsic Motivation in Reinforcement Learning

Intrinsic motivation (IM) mechanisms provide agents with internally generated rewards to motivate exploration and learning. Pathak et al. [2] suggested curiosity-driven intrinsic rewards,

encouraging agents to explore states with substantial prediction errors in a learned forward model. Similarly, Bellemare et al. [3] combined novelty detection with density-based methods, rewarding agents for experiencing novel states. Another approach is empowerment, proposed by Mohamed and Rezende [11], whereby intrinsic rewards are manufactured to maximize an agent's control over its Environment Formulating Goal Spaces from Latent Space Representations of

Raw Sensory Inputs for Autonomous Learning of Artificial Agents19environment. These mechanisms highlight the importance of intrinsic motivation in situations with sparse external reward contexts, where traditional feedback is unsuitable for learning regulation.
In an article on intrinsic motivation for reinforcement learning (RL), Houthooft et al. [7] proposed a framework involving intrinsic rewards that can guide agents by means of exploration in scenarios where extrinsic rewards are sparse or difficult to define. Their reward-based approach is meant to instill agents to find and investigate novel states, emphasizing intrinsic curiosity behavior as opposed to task-oriented reward.

While this work explicitly responds to the field by emphasizing the significance of intrinsic motivation, it is not without considerable limitations. One major drawback is the limited number of environments used in testing, which are essentially made up of comparatively simple, traversable tasks [6]. Such restricted application weakens the application of the method as agents trained using intrinsic motivation will be less effective in more realistic and practical settings [3].

Also, the strategy assumes that intrinsic rewards are enough to create sufficient exploration, perhaps neglecting other vital considerations such as long-term planning or task-specific goals, which can be even more fitting in complicated scenarios [7].The GSF(ExO) strategy tries to address these limitations by incorporating a wider number of environments, where exploration is driven not just through intrinsic motivation but also by more complex, task-dependent goals [6].
By decoupling intrinsic exploration from extrinsic task reward and learning to adapt exploration strategies, GSF(ExO) broadens the application of intrinsic motivation to more realistic and harder environments [3].

3.2 DREAM to Control: Learning Behaviors by Latent Imagination

DREAM to Control by Hafner et al. [5] is a new approach that merges model-based reinforcement learning with latent space dynamics. The main idea is to use an environment model, trained on visual observations, to hallucinate future states and roll out possible trajectories. One of the innovations in this model is the latent imagination component that enables planning and exploration with no external reward during exploration.

Instead, agents leverage latent imagination to predict future outcomes from learned dynamics to make informed decisions about future actions. While the DREAM to Control method is a significant breakthrough in the field, it is not without its limitations. One of the main challenges is that it relies on extrinsic task rewards during exploration. In environments where extrinsic rewards are absent or sparse, the agent's ability to efficiently explore and learn could be hindered. This dependence on extrinsic rewards can reinforce inefficiencies, particularly in settings where the agent should explore on its own without pre-defined goals or reward signals. This work is directly relevant to goal space design as it introduces the concept of latent space dynamics, where future trajectories can be imagined and planned without explicit contact with the real world. This concept is easily adapted to the suggestion of modeling goal-directed behaviors with latent space representations alone, as agents can explore potential goal states and identify which actions move them closer to the goals, without even real-world feedback at every time step.

Our approach is an extension of the concept of latent imagination and model-based planning, but we attempt to reduce the dependence on external rewards. We aim to create a more independent exploration mechanism that can function well in settings where task Rewards are nonexistent or untrustworthy. Through the addition of an intrinsic reward system according to the agent's power of prediction and navigation through various situations, we expect to promote greater autonomous exploration. This would help counter the limitation of sparse external rewards and facilitate improved generalizability of the agent's learning to a more diverse set of environments.

3.3 Learning Latent Dynamics for Planning from Pixels

Learning Latent Dynamics for Planning from Pixels" is an important research paper in the field of model-based reinforcement learning (RL). The authors, Ha and Schmidhuber [6], In the paper into how agents can learn a compact, latent representation of their world from high-dimensional sensory inputs, (raw pixel data) and use this representation to plan and make decisions. The paper demonstrates that agents can predict and simulate future states directly in this learned latent space, enabling them to plan complicated actions without conditioning on high-dimensional observations or needing standard, hand-designed models of the environment. What is new in this approach is that it is able to unify unsupervised learning, latent representation learning, and planning within a single framework. Here, agents learn to encode the environment's dynamics in a low-dimensional latent space, where decision-making, planning, and control are executed efficiently thereafter.

Parisotto et al. [5] take a very similar path in their model-based RL framework, learning latent dynamics from pixel-level input. Neural networks and latent variable models are integrated so as to model the environment's high-dimensional dynamics, allowing agents to plan through learned representations. Using deep learning to represent the underlying structure of the world, they show that agents can learn to carry out complex tasks with relatively few interactions.

There are, however, limits to this kind of approach, especially in terms of scalability and robustness.

Model-based algorithms often rely on fully supervised training, which can be computationally expensive and difficult to scale in very large, complex worlds. Additionally, the fidelity of the learned dynamics model has a large influence on performance. If the model is not perfect or the world provides noisy or partial observations, the approach can suffer, which limits its application to real-world domains.

To alleviate these challenges, GSF(ExO) proposes a more scalable and flexible method for learning latent dynamics. Instead of relying solely on full supervision training, our model combines supervised and unsupervised learning methods, which is more scalable. We also aim to improve robustness by offering uncertainty estimation of the learned models, allowing the agent to handle noisy or missing data more robustly.

3.4 Plan2Explore in Unsupervised World Models

In Plan2Explore, Parisotto and Salakhutdinov [9] introduce a framework that enables agents to learn a world model in an unsupervised manner. With such a world model, agents are able to plan and explore without supervision specific to tasks, only utilizing raw environmental sensory inputs.

The core idea in this framework is unsupervised exploration, where agents are forced to go out into their world without set goals or external rewards, merely to experience exploration [9].

This method is particularly relevant to the definition of goal spaces in latent space representations since it involves learning environment dynamics as well as learning about goals without external guidance. Under this framework, agents construct goals for themselves by learning to represent things in a latent space, which they can use for planning and acting. Plan2Explore also has intrinsic motivation, which praises agents for exploration that adds their knowledge of the world without necessarily relying on explicit task-based feedback [9]. A drawback of this approach is, however, the emphasis on exploration rather than on task achievement. While Plan2Explore enables agents to extensively explore their surroundings, their primary focus is optimization of exploration rather than necessarily achieving specific tasks. Therefore, the agent can invest significant effort in nongoal-driven exploration that does not contribute towards realizable goals, which can render learning less efficient and effective overall [9].

Our contribution is to enhance the Plan2Explore framework by balancing task completion with exploration. We introduce a more directed exploration mechanism where agents do not only try to explore their environment but also towards fulfilling given goals. By integrating exploration and exploitation together, we enhance the agent's learning efficiency and make its efforts at exploration worthwhile for task solving. Straight away following Plan2Explore, several other reinforcement learning approaches have explored other directions for planning, exploration, and optimal learning.

One phenomenal set of work has been in minimizing supervised learning or reward-based exploration to enable more model-based RL. Approaches such as curiosity-driven exploration, where the prediction errors of agents are used to guide behavior, have been thoroughly investigated [19, 3]. Similarly, model-free RL methods based on sparse feedback learning have

been in vogue, but this comes with the price of inefficiently slow learning and suboptimal sample efficiency for high-complexity situations.

3.5 Applications and Limitations of Current Methods

These techniques have been applied in many areas, including robotics, games, and simulated systems. GSF(ExO) works to address some of the key limitations of these techniques by offering greater generalizability, reducing dependence on extrinsic rewards, enhancing scalability and stability, and finding a balance between exploration and task completion. This technique was developed to overcome some issues in a number of key reference studies. Following is an overview of how each limitation is addressed:

Previous Work	**What's the Issue?**	**How GSF(ExO) Address It**
Intrinsic Motivation in Reinforcement Learning	It only works in a very limited set of environments, so it is hard to apply broadly.	Built to handle a wider variety of environments, making it more flexible and closer to real-world use.
DREAM to Control	This method still depends on hand crafted rewards, which is not ideal especially when rewards are rare.	Skips external rewards and uses intrinsic motivation, so the agent can still learn effectively even in sparse settings.
Learning Latent Dynamics from Pixels	It is not very scalable and can struggle in complex or changing environments.	Takes and unsupervised, scalable approach that's more robust and generalizes better to new situations.
Plan2Explore	It explores well but often misses the actual goal, which can slow down learning.	Better balance between exploring and achieving goals, leading to faster and more useful learning.

Table 3.1: Summary of Limitation on Prior Works and Corresponding Solution with GSF(ExO)

CHAPTER 4

Proposed Approach

The proposed approach allows artificial agents to automatically discover, describe, and pursue goals through the utilization of latent space representations of raw sensory inputs. The method involves constructing goal spaces, compact, abstract representations of achievable goals directly derived from the agent's learned knowledge of the world. The research approach answers three central research questions:

Q1. Can we discover goals (outcomes) that can lead to agent skill development from latent space representations of agent observations?
Q2. Can we find an intrinsic reward model that can gear autonomous learning in combination with our goal space discovery module for the agent?
Q3. Can we exploit the spatial and temporal sequences of the agent observations to improve the self-learning of the artificial agent better?

By integrating goal space with intrinsic motivational processes, agents can improve and learn autonomously, without being subject to external goals or sparse rewards. Later, with latent space representation, goal space definition, and an intrinsic reward model, the agent can learn to get the jobs done even when no objectives are explicitly defined. This approach integrates sensory input processing, learning of latent representations, and intrinsic reward mechanisms into a unifying framework and promotes more independent and autonomous behavior.

This constructed approach consists of five main components:

1. Sensory Input Processing and Latent Space Representation
2. Formulation of the Goal Space
3. Intrinsic Motivation for Goal Discovery and Learning Performance
4. Policy Optimization in the Goal Space
5. Training and Evaluation Framework

4.1. Sensory Input Processing and Latent Space Representation

The initial operation of the suggested framework is the transformation of raw sensory input, namely high-dimensional visual inputs of the DM Control Suite, to accept concise latent representations.

This transformation enables the agent to navigate complex worlds with less effort through dimensionality reduction without compromising important task-specific information. Through encoding valuable features and ignoring redundancy and noise, the latent representations facilitate better decision-making.

To learn these hidden spaces, the model employs self-supervised or unsupervised learning, i.e., Variational Autoencoders (VAEs) [6]. These representations learned are employed as a foundation for goal discovery and self-directed decision-making, hence optimizing learning and exploration.

4.1.1 Data Pipeline

Sensory Input Processing

Raw sensor readings or environment observation images are fed into a Variational Autoencoder (VAE) to learn a compact latent representation. The VAE's encoder takes such high-dimensional inputs and maps them to a structured latent space in a way that preserves important features necessary for task-relevant abstraction. These are the raw sensory input types that are accepted by the agent:

- Images: High-dimensional visual information obtained through a virtual camera.
- Proprioceptive Information: Joint angles, velocities, or accelerations of internal states.
- Environmental States: Forces, velocities, and positions of external objects outside the system

Preprocessing as well as Normalization

To deliver enhanced stability in addition to successful learning of representations, sensory inputs are subjected to the following preprocessing:

1. Normalization: Inputs are normalized to a constant range ([0,1]) for more efficient learning.

2. Feature Extraction: Simple filtering operations such as image edge detection emphasize the significant features.

3. Time-Stacking: For temporal understanding tasks, sequence frames or states are stacked to learn motion dynamics.

Latent Space Representation Learning

The VAE is trained to maximize the latent space for interpretability and disentanglement [6]. The structured representation creates a foundation for policy learning and goal formulation by:

- Preserving Critical Features: Preserving task-critical information, (an agent's position, velocity, or balance).
- Dimensionality Reduction: Bypassing high-dimensional inputs to lower-dimensional representations for effective policy learning.
- Helping Goal Representation: Enabling smooth interpolation between different states for effective generation of goals.
- Exploration Facilitation: Encouraging agents to explore new regions in the latent space, allowing for productive goal discovery.

4.1.2 Variational Autoencoder Architecture

It utilizes a VAE to learn a disentangled and compressed latent space with less than minimal information loss during encoding, as a way of rendering the observations interpretable.

Architecture Overview

The VAE used in this process is one that is responsible for learning visual data specifically, 64×64×3 RGB images from the environment of the agent. The VAE is built following the typical encoder–decoder architecture, which helps the agent to encode images into a lower-dimensional, more representative (or latent vector) representation and reconstruct them.

Component	**Description**
Input	64x64x3 RGB images
Encoder	4 Conv layers (32 to 256 filters, stride 2, ReLU)
Latent Space	32-dim vector (mean μ, log-variance log(σ2))
Decoder	4 transposed Conv layers + sigmoid output
Loss	ELBO: reconstruction loss + β * KL divergence (β = 4)

Table 4.1: Architecture Overview

Latent Sampling: For getting a useful latent vector z, the VAE employs the re-parameterization trick:

$$z = \mu + \sigma \cdot \epsilon \quad \text{where } \epsilon \sim \mathcal{N}(0, I)$$

…….4.1

Where: μ shifts the center of the latent space.

σ scales the size of the latent space.

ϵ introduces randomness from a normal distribution. This renders the model capable of sampling z while being trainable with gradients.

Decoder: Decoder takes this z vector and transforms it, inputs it into a fully connected layer and then applies four transposed convolutional layers (essentially reversing the encoder). The output layer applies a sigmoid activation to generate the output image, from 0 to 1.

Training Goal: The VAE learns to reconstruct the input image correctly along with making the latent space well-structured and continuous. To do so, it optimizes a loss that goes by the name of the Evidence Lower Bound (ELBO):

$$\mathcal{L} = \mathbb{E}_{q(z|x)}[\log p(x|z)] - \beta \cdot D_{\mathrm{KL}}[q(z|x) \parallel p(z)]$$

……4.2

Where:

- The overall loss (ELBO)
- The expectation operator,
- First term: Checks the accuracy to which the model is able to reconstruct the input.
- Second term: Keeps the latent space clean (close to normal distribution).
- β: Controls how much we desire to disentangle the latent space. input. KL Divergence.

Forces the latent space to be distributed normally, making it easier for the agent to generalize. We use a β value of 4 (used in the β-VAE approach), which helps the model

learn disentangled and explainable features essential in designing well-defined objectives in latent space. The latent space z is continuously updated online via a replay buffer of 1M samples and trained with Adam optimizer (lr=1e-3, batch size=32).

4.2 Formulation of the Goal Space

In reinforcement learning, particularly for autonomous agents, it is necessary to formulate goals in a manner that the agent will understand and be able to act on. Rather than dealing directly with high-dimensional sensory inputs such as raw images, goal space formulation converts such inputs into a structured, lower-dimensional representation that embodies the task's essence.

This transformation is typically performed by a Variational Autoencoder (VAE), an unsupervised learning model that translates high-dimensional sensory observations into compact latent vectors [5]. It is these latent vectors, representing abstracted and meaningful environmental features, that serve as the basis for goal specification.

4.2.1 How the Agent Makes Use of Latent Representations

Rather than acting directly upon raw images, the agent acts within this latent space. That is how the process unfolds:

- Input Processing: The agent perceives its world as raw sensory input.
- Encoding: These are fed through a VAE, which encodes them onto a latent state - a concise encoding of task-relevant features.
- Goal Definition: The goal is formally a target point (vector) in this latent space. The goal of the agent is then to minimize the "distance" between its current state and this target in latent space, usually under a Euclidean or cosine similarity metric [39].

The Pipeline for Goal Space Formulation

- Latent Data Collection: As the agent is interacting with its environment, it is collecting a dataset of latent representations: D={z1,z2,.,zn}
- Clustering for Goal Discovery: The agent groups these latent representations into k clusters through techniques like k-means clustering. The centroid of each cluster is a prototype for a potential goal, capturing shared and distinctive states from experience.

- Policy Conditioning: Once a goal zgoal is chosen, it conditions the agent's policy. The policy π(atzt,zgoal) prescribes which action to take at time t, guiding the agent from its current state zt toward the goal.

4.2.2 Latent Space Goal Discovery

Goals are triggered automatically by grouping areas of the latent space that generate novelty or competence increase [6]. The clustering algorithm proposes potential goal states from exploration data [5]. With intrinsic motivation signals, the agent chooses candidate goals from the latent space.

This entails:

a. **Clustering Latent Representations:** The agent groups latent states z into useful clusters, identifying areas of interest in the latent space. Clusters can be goals, having steady running velocities or steady upright pole angles [5].

b. **Goal Sampling:** Goals are dynamically sampled from the latent space to enhance diversity and adversity. Sampling is beneficial to areas that offer possibilities for ample learning or novelty [4].

c. **Dynamic Goal Adjustment:** The agent dynamically adjusts goals through intrinsic reward feedback so that learning is continuous and adaptive [3]. Having established goals, the agent learns policy in an attempt to achieve them. These policies operate within the space of goals using reinforcement learning algorithms like Proximal Policy Optimization (PPO) [7]. The agent enhances its knowledge about the environment and performance through iteratively seeking, setting goals, and maximizing action. PPO is used to enable the agent to learn policies for reaching the set objectives [6]. The policies are conditioned on the agent's state and objectives in the latent space [7].

4.3 Intrinsic Motivation for Goal Discovery and Learning Performance

Intrinsic motivation is central to autonomous learning as it enables agents to learn driving towards self-meaningful goals without the guidance of reward definitions from the world. This subsection describes how intrinsic motivation supports goal discovery and accelerates learning through exploration encouragement, goal detection in the latent space, and optimization of behavior due to novelty and signals of making progress. Our approach involves the use of

intrinsic motivation as a main driver towards successful exploration and self-learning, primarily in environments, the DeepMind Control Suite (Cheetah Run, Cartpole Swing-Up, and Hopper) [4].

Task	**Latent Goal (z_goal)**
CartPole Swingup	Upright pole with zero cart velocity
Cheetah Run	Maximum forward velocity
Hopper	Stable hop with forward movement

Table 4.2: Goal Representation by Task

4.3.1 Intrinsic Reward Mechanism

Intrinsic rewards are calculated based on both goal learning and agent learning. The two are combined, balancing exploration and task achievement. Including these factors, intrinsic reward helps to enhance the agent's learning process.

The overall intrinsic reward R(z,t) at state z at time t is weighted together of the learning performance reward and the goal learning reward. The weights w1 and w2 resize the two terms in relation to each other as per their relative weighting:

Rtotal Intrinsic Reward: Total intrinsic reward at time t is a weighted sum of these two terms:

$$R_{\text{total}}(t) = w_1 \cdot R_{\text{goal discovery}}(t) + w_2 \cdot R_{\text{learning progress}}(t)$$

…….4.3

Where:

• Rtotal(t) is total intrinsic reward for at time t

• w1 and w2 are hyperparameters that control the exploration (novelty) vs. exploitation (progress towards ability) trade-off. By maintaining a balance between exploration and development of skill, we encourage the agent to learn important goals and to develop its representations in an economical and stable fashion.

4.3.2 Goal Discovery Reward

Goal discovery encourages the agent to venture into new regions of the latent space, which are less visited or never visited at high frequency. This reward component encourages the agent to visit new states and goals, that are followed by more exploration. Intrinsic motivation is an autocatalytic process, as it encourages exploration and goal discovery in situations where there are no or scarcely any external rewards. The primary objectives of intrinsic motivation are:

• **Exploration drive:** Encourages the agent to explore new parts of the latent space, leading to new state or behavior discovery. For instance, in Cartpole Swing-Up, intrinsic drive gets the agent to try different angular velocities in an attempt to get the pole up successfully [4].

• **Discovery of Significant Goals:** The agent discovers goals in the latent space by selecting states that are both reachable and diversified, related to different behavior or accomplishments. In Cheetah Run, significant goals can be reaching various speeds with constant [4].

• **Exploration vs. Exploitation Trade-off:** Intrinsic motivation is employed to weigh finding new goals against exploiting existing goals that enhance the performance of the agent. The novelty of a state z is measured by the degree to which it is "unvisited", generally the distance to the previously visited states in the latent space. A straightforward novelty measure is the minimum distance from the current state z to the previously visited ones, [5]. It promotes exploration and goal learning as the agent is rewarded for visiting new, unvisited regions.

$$R_{\text{goal discovery}}(t) = \alpha \cdot \text{Novelty}(s_t) \quad \text{.....(4.4)}$$

Where:

• α is a scale factor = range 0.1 to 1,

• st's is the latent state, and

• Novelty is an estimate of closeness to explored states.

4.3.3 Learning Progress Reward

The competence reward internally ensures that the agent is making measurable progress towards goals. Through the emphasis on improvement over time and adjustment in response to performance, this reward promotes agents to not only explore new areas (by curiosity) but also to focus on the enhancement of their task execution. Through balance, exploration and good exploitation of known knowledge can be achieved. The learning progress reward thus becomes a

major element of the agent's intrinsic motivation, which promotes continuous improvement in the execution of tasks.

Reward for Improvement: The student receives a reward for improving significantly to reach some target, reducing errors or the performance measure in a series of attempts. Improvement is measured as a performance measure, ΔP, over the interval. Learning progress reward is the absolute difference between the performance measure at time t and the previous time step, t-1, performance.

$$R_{\text{LP}} = |P_t - P_{t-1}| \quad \text{.....4.5}$$

Where:

- RLP is the learning progress reward.
- Pt is the measure of performance or progress at time t.
- Pt−1 is the measure of the previous time step's performance or progress.
- Absolute difference makes the reward function sensitive to the size of the progress, in the positive or negative sense.
- The magnitude Pt − Pt−1 captures how the agent's capacity to attain what it desires has improved or worsened.

An excellent disparity signifies improvement or worsening considerably, which in turn elicits rewards or punishment. For example, in Cheetah Run, improvement in learning is reflected through improvement in the agent's running speed or balance across episodes. Learning performance reflects the agent's performance in becoming better and executing a task in any state or goal. It can be measured by competence indicators that monitor the agent's performance in task completion over time.

The competence of the agent C(z,t) at any hidden state z at time t can be estimated using a measure of performance, such as the success rate or the accuracy of task completion:

C(z,t) = Successful attempts at z / Total attempts at z.

Improvement in learning is the enhancement of competence over time, and mere improvement in competence is what qualifies as part of the agent's improvement. Therefore, the internal reward of the learning improvement, Rlearning improvement (t), is calculated from the improvement. 34

$$R_{\text{learning progress}}(t) = \gamma \cdot \left(\frac{\Delta \text{performance}}{\Delta t} \right)$$

…….4.6

Where: ΔPerformance is the difference between time episodes. It is the extent to which the agent's \skills change from episode to episode, rather than at every infinitesimal step in an episode. It measures progress on large learning intervals, not on small transient changes.
Δt is time elapsed.

4.3.4 Combined Intrinsic Reward

This strategy merges intrinsic motivation with task-specific goals into a single learning framework in such a way that agents are motivated to venture into new states as well as solve tasks and achieve their objectives. Weighted reward allows for a mechanism to control the priorities of the agent's learning, which is of paramount importance in achieving effective and efficient learning in sophisticated environments. The discounted cumulative reward, J(π), is approximated by the sum of the intrinsic rewards in each time step, each of them discounted by a factor γ (discount factor which assigns more weight to closer rewards than to far ones).

$$J(\pi) = \mathbb{E}\left[\sum_{t=0}^{T} \gamma^t R_{\text{total}}(t) \right]$$

…….4.7

Where:

- T denotes the time horizon
- γ is the discount factor, between 0 and 1, which represents how much future rewards are preferred over immediate rewards.

4.4 Policy Gradient Update and Optimization with Intrinsic Reward

To improve its behavior, the agent uses policy gradient methods to adjust its decisions based on how much intrinsic reward its actions generate. Over time, it learns to favor actions that lead to more learning or discovery. In tasks like Cheetah Run and CartPole Swing-up, this helps the agent strike a balance between exploring new strategies and refining what works gradually improving through experience to maximize long-term intrinsic rewards.

$$\max_{\pi} \mathbb{E}_{z\sim\pi} \left[\sum R(z_t, t)\right]$$

……4.8

Where:

- π is the policy being trained.
- Ez π is the expectation over z trajectories sampled from policy π.
- R(zt,t) is the reward at time t in a state or trajectory zt.
- The overall cumulative reward over time is represented by the summation ∑R(zt, t).
- Maximize this cumulative expected reward by finding the policy π.

The agent is driven to explore unfamiliar states (novelty) and improve its skills (learning progress). It earns intrinsic rewards from both pushing it to try new things and get better over time.

By combining these two signals, the agent is encouraged to explore meaningfully and grow more competent. This combined reward guides its learning, as shown in the formula below:

$$R(z,t) = w_1 \cdot \alpha \cdot \text{Novelty}(z) + w_2 \cdot \beta \cdot \Delta + C(z,t)$$

……4.9

Where:

- novelty(z): Measures how novel or less frequently visited the latent state z is. Building
- Δ: a measure of how much the agent's performance has improved.
- c(z,t): used to implement task-specific regulation or environmental feedback.
- α, β: Scales for novelty and learning progress respectively.
- w1, w2: Balance weights for contribution of each term to the overall intrinsic reward.

This cumulative intrinsic reward guides the agent to learn and explore well even without external supervision. The agent's policy π determines the probability of the actions from its state, i.e., π(atst). The goal is to maximize this policy to produce maximum expected cumulative intrinsic reward in the long term. This optimization process is bringing a balance between venturing into new, uncharted terrain (novelty) and growing its proficiency (competence), resulting in enhanced and autonomous learning in complex situations.

4.5 Policy Optimization with Intrinsic Reward

The intrinsic motivation signals are integrated into the reinforcement learning framework to fine tune the agent's policy $\pi(atzt)$, helping it achieve its goals. Proximal Policy Optimization (PPO) is used to train the policy, ensuring that updates remain stable and reliable. These intrinsic rewards steer the agent toward exploring new goals or enhancing its task performance.

Integration with MDP Framework

The agent operates in an MDP defined as (S, A, P, R, γ)

- **State (s)**: The raw sensory inputs from the Hopper environment (joint angles, velocities, and torso position) represent the state st at any time t.
- **Action (a)**: The agent selects actions at, that control the torques applied to each joint (hip, knee, ankle).
- **Transition Function (P)**: The environment responds to the agent's actions by updating the state, the agent applies torques to joints, causing the hopper to move and its state (position, velocity, etc.) to change according to the physical dynamics of the system.
- **Reward (R)**: The agent receives rewards based on its performance, such as a reward for maintaining balance or a penalty for falling over.
- **Discount Factor (γ)**: Determines how much the agent values future rewards when learning

Parameter	**Value**
Batch Size	32
Learning Rate	1e-3
Epochs per Episod	50–100

Table 4.3: PPO Training Hyper parameters

4.6 A Structured Approach to Training a Reinforcement Learning Agent

Successful training of an agent in a reinforcement learning (RL) setting must be accomplished by a systematic design in order to ensure the agent learns properly and achieves the desired behavior.

Training of an agent in a reinforcement learning (RL) setting is best accomplished by a systematic approach in order to enable the agent to learn efficiently and achieve the desired behavior. While we go through these steps and apply the parameters provided to us, we would be able to effectively check the autonomous learning ability of the agent and make the training process for tasks efficient and stable. Below is a step-by-step process: The whole system is tested on three benchmark environments:

Step	**Description**
1	Agent explores environment, collects observations
2	VAE encodes into latent space
3	Goal space generated from clustered latents
4	PPO optimizes behavior toward z_goal
5	Evaluate using task-specific metrics

Table 4.4: Process Steps with Description

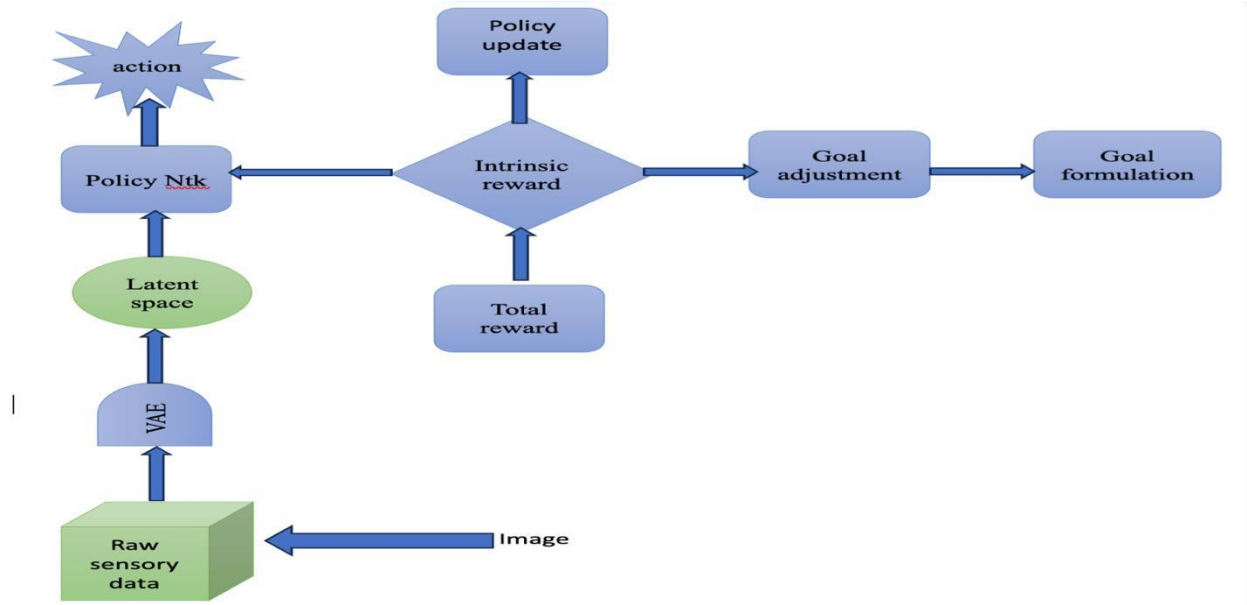

Figure 4.1. Block diagram of Proposed Approach

4.6.1 Environment Setup

We use the DeepMind Control Suite (DM Control Suite) as the baseline to compare our proposed methodology [8]. The suite offers high-quality, physics-based environments tailored to continuous control tasks and is therefore a good environment to implement agents that are trained to build goal spaces from latent space representations of raw sensory inputs [9]. The DeepMind Control Suite is a collection of various continuous control tasks that were designed to assess the proficiency of reinforcement learning (RL) agents in controlling complex systems with minimal supervision [8].

The common tasks of CartPole Swing-up, Cheetah Run, and Hopper are highly utilized in deep reinforcement learning (DRL) research to test the ability of agents to learn and apply sophisticated control schemes [9].

4.6.2 Agent Selection

We focus in this paper on allowing artificial agents to automatically learn goal spaces from latent space representations of raw sensory input. We seek to determine how agents can learn to

perform tasks on their own in tasks like CartPole Swing-up, Cheetah Run, and Hopper—without knowledge of the task in advance or extrinsic reward, and with nothing but latent representations to guide learning. The agent uses a combination of essential components to facilitate goal-directed autonomous learning:

a) **Reinforcement Learning Framework:** Agents learn policies in an MDP via PPO (Schulman et al., 2017), which is appropriate for continuous control in environments like the DeepMind Control Suite.

b) **Goal-Conditioned Learning:** Policies are conditioned on latent goal representations, enabling agents to aim for desired results in latent space (Andrychowicz et al., 2017).

c) **VAE Integration:** Variational Autoencoders (VAEs) compress high-dimensional sensory input into lower-dimensional latent spaces, without sacrificing task-relevant information (Kingma & Welling, 2013).

CartPole Swing-up: The agent learns to swing and balance a pole upright from raw observations (position, velocity) compressed into latent representations.

- **Latent Space Function:** Latent space codes dynamics of pole angle and cart position, allowing for the learning of control strategies for balancing.
- **Without Reward:** the agent learns motion patterns driven by prediction error minimization to balance the pole.
- **Autonomous Learning** is realized by repeated interaction and motor policy refinement within latent space.

Cheetah Run: In this task, the agent has a robotic cheetah run in a simulation, learning to coordinate its legs and adapt its movements to run successfully.

- **Latent Space Function:** Latent encodings contain limb dynamics (velocities, positions), allowing the agent to discover locomotion patterns (sprinting, walking).
- **Without Reward:** In the absence of any explicit reward, the agent discovers movement in the latent space in patterns that equate to stable locomotion. The agent is able to discover movement strategies.
- **Raw Sensory Input:** The agent is fed proprioceptive information (joint angles, velocities) and possibly visual input (if operating in a more complex environment).

- **Autonomy Learning:** The agent learns to run in a smooth manner by figuring out how to move its limbs from complex sensor data. With no rewards from the outside, it employs curiosity, generating different movements and learning by trial and error to improve over time.

Hopper: This setting involves a simulated 2D robot (the "hopper") that must learn to hop forward on a flat plane. The agent must balance the body, manage leg movements, and coordinate joints to move efficiently.

- **Latent Space Function:** The agent compresses joint angles and velocities into latent states in order to identify configurations that facilitate stable hopping.
- **Without Reward:** The agent is able to learn to find its body configuration space through curiosity to determine the states that lead to more effective or stable hops. Through internally generated rewards, the agent learns its hopping ability without any external reward.
- **Raw Sensory Input:** The hopper robot uses proprioceptive data, i.e., positions, angles, and velocities of leg joints, to sense its body state.
- **Autonomy Learning:** The agent is taught to hop by attempting different leg and body positions via raw sensor input. Driven by curiosity, it learns to hop effectively by predicting and improving where it will land with a hop, without any external rewards.

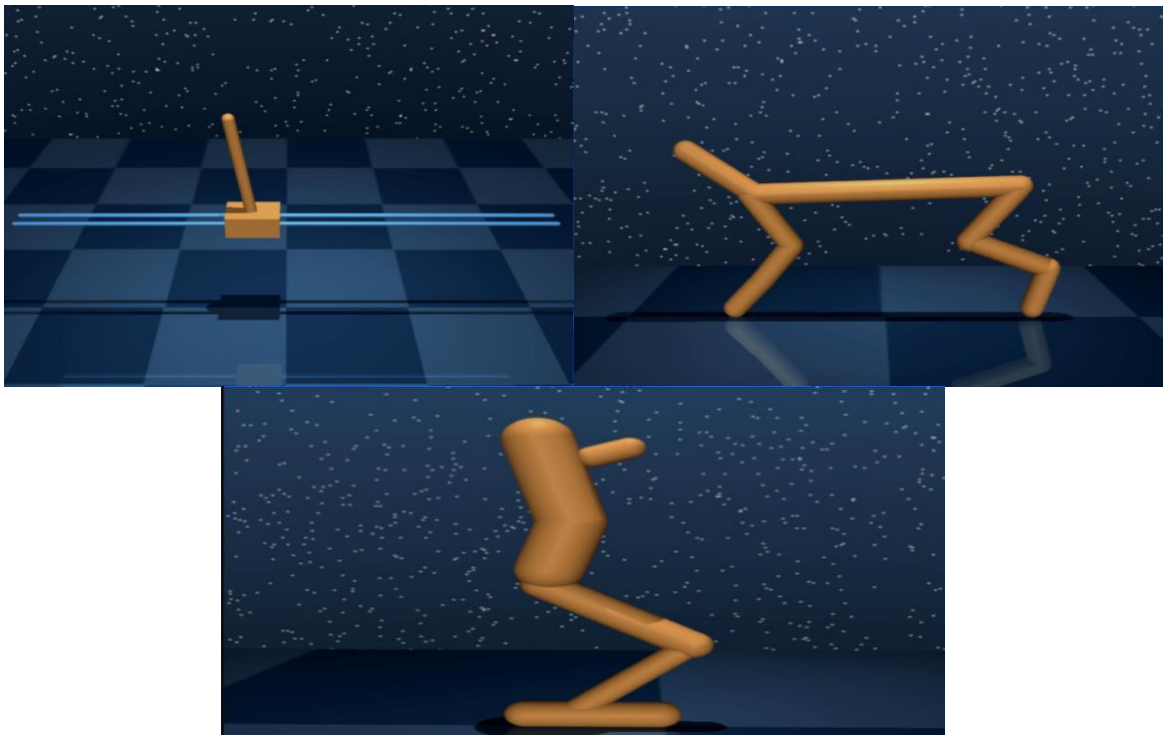

Figure 4.2 Images of Deep mind Control Suit Agent Hopper, CartPole Swing-Up, and Cheetah Tasks

1. Formalize the MDP (Markov Decision Process)

State Space (S): Define the observations that the agent receives, such as joint angles, velocities, torques, and contact states.

Action Space (A): Specify the range and type of actions the agent can take, for example, continuous torques applied to joints.

Reward Function (R): Design a reward function to incentivize desired behaviors, like moving forward, maintaining balance, or achieving energy efficiency. Here, we incorporate intrinsic rewards like curiosity, exploration, or progress in the latent goal space.

Transition Function (P): Ensure the environment accurately models the dynamics, describing how actions lead to state transitions.

Discount Factor (γ): Set the discount factor to balance the importance of short- term versus long term rewards.

2. Select the RL Algorithm

Choose an appropriate reinforcement learning (RL) algorithm. In this case, we select Proximal

Policy Optimization (PPO) based on its suitability for model-free algorithms.

3. Data Preparation

- Preprocess Observations: Normalize state inputs to standardize the data.
- Reward Scaling: Clip or normalize the rewards in order to stabilize the gradients during training.
- Action Clipping: Restrict the actions to plausible ranges (torque limits) to prevent instability.
- Replay Buffer: to store previous trajectories from the current policy for a number of training epochs.

4.6.3 Training Process

1. **Start by Initializing the Agent:**
   - Randomly set up the agent's behavior policy at the beginning.
   - Set up exploration strategies, like epsilon-greedy, to encourage the agent to explore different actions during the early stages of training.
2. **Interact with the Environment:**
   - At each step, the agent observes its current situation (state).
   - It picks an action based on its current understanding of the environment (policy).
   - The agent then receives feedback in the form of a reward and the new state it finds itself in.
   - This experience, consisting of the current state, action taken, and reward received, and the next state, is saved in the replay buffer to be used for future learning.
3. **Update the Agent's Policy:**
   - Periodically, the agent picks a random batch of experiences from the replay buffer.
   - It processes these experiences to compute gradients and then updates both its policy and
   - value models based on the new information.
4. **Keep Track of Progress:**
   - Regularly monitor key performance indicators, the average reward, the length of episodes, and overall stability in how the agent is performing.
5. **Adjust Exploration Over Time:**

- As the agent gets better at the task, gradually reduce how much it explores. This can be done by lowering the exploration noise (like reducing epsilon in epsilon-greedy), allowing the agent to focus more on exploiting what it's learned.

6. **Begin the Training Process:**

- The agent begins to interact with the environment.
- Occasionally, the agent will sample batches from this buffer and use them to learn to play the game better using something called an algorithm like PPO.
- Keep track of important metrics like the average reward per episode or the agent's performance over time.

7. **Test the Agent's Abilities:**

- Test how well the agent performs by removing the exploration phase, meaning no random actions, just what it has learned so far.
- Evaluate its performance using metrics like average reward over the last 20 episodes or the success rate. Test the agent under slightly different conditions (different terrain or unexpected disturbances) to see how well it generalizes.

8. Optimize the Training

Fine-tune key training settings across various agents (CartPole, Cheetah, and Hopper) to improve stability and performance:

- Learning Rate: Start with 1e-3, adjusting as needed.
- Batch Size: Start at 32, experimenting with sizes from 32 to 128.
- Discount Factor (Gamma): Set at 0.99 to encourage long-term rewards.
- Timesteps: Train for 1 million steps.
- Replay Buffer Size: 1 million to ensure the agent gets enough experience.
- Entropy Coefficient: 0.05 to balance exploration and exploitation.
- Observation Size: 64×64×3 RGB images for the agent's input.
- Episode Length: Cap each episode at 1,000 steps.
- Action Repeat: Repeat each action twice (R=2) to smooth the training process.

9. **Log and Analyze Results:**

We use TensorBoard to visualize how the agent's performance changes over time, helping you identify if there are any plateaus or divergences in training.

10. **Monitor and Adjust Progress:**

Monitor how the agent's performance (Reward Return) progresses during training. If our performance plateaus or oscillates a lot, we adjust either the learning rate, the batch size, or the entropy coefficient to create stability in learning.

11. **Fine-Tuning the Model:**

After the initial training, fine-tune key parameters like the learning rate or entropy coefficient based on how well the agent is performing, making sure it's still improving.

12. **Deploy and Iterate:**

- Once the agent performs well, deploy it for its intended use case.
- Collect feedback from the deployment to see how it performs in real scenarios and retrain the agent if needed, incorporating new data or situations.
- Adjust the reward function or exploration strategy if the agent's performance starts to degrade or can be improved.

4.6.4 Evaluation Metrics

Performance assessment of reinforcement learning (RL) agents is crucial to evaluate how well they learn, generalize and solve tasks. For instance, tasks such as Hopper, CartPole Swing-Up and Cheetah on DeepMind Control Suite have all different characteristics, where performance and sample efficiency need to be very carefully measured. The evaluation metrics we use to measure RL agents trained for these tasks in this paper are:

- Cumulative Reward: The sum of reward since the beginning of an episode.
- Average Reward per Episode: Average reward gained by the agent in an Episode, The former score is indicative of how well it performed overall.
- Success Rate: The fraction of episodes the agent successfully accomplishes the task.
- Sample Efficiency: How many data points, samples or interactions the Agent needs in order to reach a certain level of performance.
- Learning Speed: How well the agent learns over time. These quantities are of importance for measuring the exploration and learning capability of an agent to optimize its policies in dynamic and continuous control spaces.

Cumulative Reward:

Cumulative reward is one of the most fundamental metrics in reinforcement learning. It represents the total rewards collected by the agent during an episode and reflects the overall success in achieving the task objectives.

$$R_{\text{cumulative}} = \sum_{t=0}^{T} r_t$$

.....4.10

Where:

• Rcumulative represents the total accumulated reward over time.

• rt is the reward received at time step t.

• T is the final time step.

The summation runs from t=0 to T, summing up all rewards.

Average Reward per Episode:

The average reward per episode provides a more stable and reliable evaluation by calculating the mean reward collected across multiple episodes. It helps to mitigate the effects of variability inindividual episodes.

$$R_{\text{avg}} = \frac{1}{N} \sum_{i=1}^{N} R_i$$

.....4.11

Where:

- Ravg represents the average reward.
- N is the total number of rewards.
- Ri is the reward at index i
- The summation runs from i=1 to N, averaging all rewards.

Useful for comparing agents' performance over many trials and helps identify consistent improvements or regressions in training. This metric highlights the agent's ability to perform consistently, rather than relying on occasional high-performance episodes.

Success Rate

The success rate measures the percentage of episodes in which the agent achieves a predefined goal or meets a success criterion, such as balancing for a certain time or traveling a minimum distance.

$$\text{Success Rate} = \frac{\text{Number of Successful Episodes}}{\text{Total Episodes}} \times 100\ldots.$$

….....4.12

This metric provides a binary measure of task achievement and is particularly useful for tasks with well-defined goals.

a) CartPole Swing-Up

The agent is considered successful if it can keep the pole balanced upright for a certain amount of time without letting it tilt too much. In simple terms, this means the pole must stay within (±15° of the vertical for a set number of time steps. If it manages to do so, the agent has effectively learned to maintain balance.

$$\sum_{t=0}^{T_{\text{success}}} \mathbf{1}(|\theta_t| \leq \theta_{\max}) = T_{\text{success}}$$

….....4.13

Where:

- θt is the pole's tilt angle at time t,
- θmax is the allowed tilt (15° or 0.26 radians),
- Tsuccess is the required time duration for success,
- 1() is a function that counts how many time steps the pole stays within the angle limit.

b) Cheetah Run

For the Cheetah agent, success implies being able to run at a steady pace for a period of time. That is, it needs to reach and maintain a minimum speed of 3 m/s for a set number of time steps. If it can do that, then this could mean it has successfully learned the task.

$$\sum_{t=0}^{T_{\text{success}}} \mathbf{1}(v_t \geq v_{\min}) = T_{\text{success}}$$

….....4.14

Where:

- Vt is the Cheetah's speed at time t,
- vmin is the speed threshold (3 m/s),

- Tsuccess is the required time duration,
- Function 1() counts the time steps where the speed is high enough.

c) Hopper

Success means hopping forward a specific distance without tipping over for the Hopper agent. To

achieve this, the agent must cover at least a set distance (5 meters) while staying upright for a certain amount of time.

$$d_T \geq d_{\min} \quad \text{and} \quad \sum_{t=0}^{T_{\text{success}}} 1(\text{not fallen}) = T_{\text{success}} \qquad \text{….....4.15}$$

Where:

- dT is the total distance traveled,
- dmin is the minimum required distance (5 meters),
- The second condition ensures the agent doesn't fall during the required time steps.

Sample Efficiency

Sample efficiency measures how well a reinforcement learning (RL) agent learns from a limited number of interactions with its environment. This is important for tasks where collecting data is expensive or takes a lot of time. It tracks how quickly the agent improves its overall performance based on the number of steps it takes in the environment.

An agent with higher sample efficiency can learn faster and more effectively. This makes it especially valuable in real-world situations where data is scarce or costly to obtain.

Learning Speed

Learning Speed refers to how quickly an agent reaches a certain level of performance or finds the best possible strategy. While it's closely related to sample efficiency, learning speed is more about the time it takes to reach that point, rather than how many samples the agent uses. It's measured by how many training steps the agent needs to achieve a specific goal, like hitting a target reward or success rate.

4.7 Methodological Considerations

Category	Details
Data Collection	Random exploration phase collects heterogeneous sensory information.
Analysis Procedures	Latent clustering using k-means. Reward shaping from novelty and progress metrics. Policy tuning via cumulative intrinsic reward.
Rationale	VAE: Chosen for generative, interpretable latent structure PPO: Stable in continuous control and efficient with sparse rewards
Justifications	Latent representations are semantically consistent. Intrinsic signals are sufficient for learning.
Limitations	Goal ambiguity in the latent space

Table 4.5: Methodological Considerations

CHAPTER 5

Result and Discussion

Our work stands apart within this literature by presenting an approach of dynamically and unsupervised shaping goal spaces in robust latent spaces from raw sensory inputs. Some differences compared with previous works, our adaptive intrinsic reward method uses both latent goal novelty and uncertainty, which allows scalable exploration in high-dimensional and non-stationary environments [1].

In this section, we test the performance of our method on multiple benchmark tasks and compare with other two state-of-the-art methods Plan2Explore and Dreamer. We experiment on the tasks of Cartpole Swing-Up, Cheetah Run and Hopper, which are different in the difficulty of continuous control problem and autonomous learning problem. Both required the agent to interpret challenging sensory inputs, maximize its policies, and generalize over levels of difficulty. We compare methods on key performance metrics including average reward, learning speed, sample efficiency, and success rate. They will give a full picture of how well the agent can learn effectively, adapt to task dynamics and achieve effective behaviors. We use same hyperparameters for all the environments.

The conversation demonstrates the fact that GSF(ExO) always performs better than or as well as common approach in learning efficiency and task reporting rewards. It also indicates the positive effects on learning robustness and scaling by combining intrinsic rewards and the formation of the latent space of goals.

5.1 Experimental Setup

Experiments are performed on the DeepMind Control Suite with high-dimensional sensory inputs and a range of environments. One of the fundamental problems is Navigation, which concerns letting the agent navigate through 2/3D environments to certain target locations and uses visual observations to guide the agent's movements. The observations are the RGB images of the resolution of 64×64×3 pixels.

The learning of task-specific features from the input data via fine-tuning requires an end-to-end differentiable network and there is a tradeoff between the amount of features that are learned and the number of required labels (Kornblith et al., 2019). The setup, this makes the agent completely depend on the visual input, just as in real-life where the agent needs to make sense of raw data (images) to construct useful representations for solving the task.

5.1.1 Initial Training Runs

We first train the agent with the specified hyperparameters for 1 million time steps, and monitor its performance closely.

5.1.2 Adjust Hyperparameters

We update the batch size, learning rate, or entropy coefficient according to the patterns of learning curves while the agent learns. These adjustments further optimize the training process to the specific task at hand.

5.2 Performance Evaluation

We assessed each of above methods, including Plan2Explore, Dreamer and GSF(ExO), with the following core metrics:

- Mean reward (success in the task)
- Learning rate (episodes to convergence)
- Sample efficiency (reward to environment interaction ratios)
- Accuracy (percentage of successful task completions)

Although we provide visual comparison against the metrics, we focus more on interpretive analysis to reveal the reasons behind those metrics performance differences.

5.2.1 Performance on Cartpole Swing-Up Task

The Cartpole Swing-Up is also considered a benchmark to evaluate the ability of reinforcement learning agents to do balance and control. In this work, we compare Plan2Explore, Dreamer, and

GSF(ExO) by the performance in key metrics such as average reward, sample efficiency, success rate, and learning speed. The focus is on how each approach addresses this dynamic control problem, focusing on the advantages GSF(ExO) offers for formulating a goal space.

Average Reward

The average reward reflects the agent's performance during evaluation tests in CartPole Swing-Up task, the higher the average reward the better. As can be observed in Figure 5.1, we found the following trends when comparing the average rewards of the different methods:

- **Plan2Explore (803.53):** This approach has the lowest average reward, and we conjecture that it does well, but its exploration or policy will not fully capitalize on task-relevant visual information to yield higher rewards.
- **Dreamer (826.07):** Dreamer has only slightly better performance than Plan2Explore, with a minor improvement of better utilization of model-based reinforcement learning and visual features for policy learning.
- **GSF(ExO) (874.84):** GSF(ExO) generates the best average reward, outperforming Plan2Explore and Dreamer. This means that GSF(ExO) is more efficient in handling visual data, presumably as a result of reinforced representation learning, better exploration strat- egies, or more advanced policy optimization.

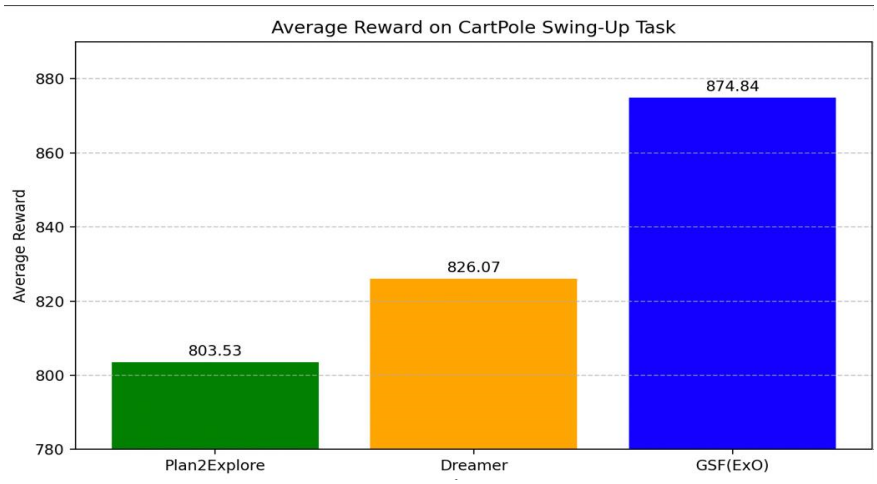

Figure 5.1: Average Reward Comparison on Cartpole-Swing-Up Task

The attained results demonstrate that the obtained solutions increase slowly according to three tested methods, of which, GSF(ExO) is the largest. The small gap of 71.31 points between the scores of Plan2Explore and GSF(ExO) shows that all methods are able to solve the task to some extent. But GSF(ExO) has a superior fine-tuning and policy refinement, so it obtains the highest average reward.

Learning Speed

Basically, learning speed refers to how fast each does method catch on to things during training. In this instance, we want to know how long it takes an agent to learn the swing-up task in cartpole.

- **Plan2Explore** takes the longest to converge, requiring approximately 2000 episodes to start performing well. It feels like a lot of time is spent exploring, but not necessarily in the most helpful ways early on, which slows down the action.
- **Dreamer** moves a tad bit faster, arriving at it in roughly 1500 episodes or so. It appears that it benefits from being able to anticipate what might occur next, enabling it to learn more quickly than Plan2Explore, but not instantly.
- **GSF(ExO)** is the fastest among the three. Though it converges to a steady performance after only one thousand episodes, demonstrating the it's faster learning. That means it's better at focusing on the right information and making smart decisions early in training.

In Figure 5.2, GSF(ExO) has faster learning speed than Plan2Explore. Dreamer is somewhere in between.

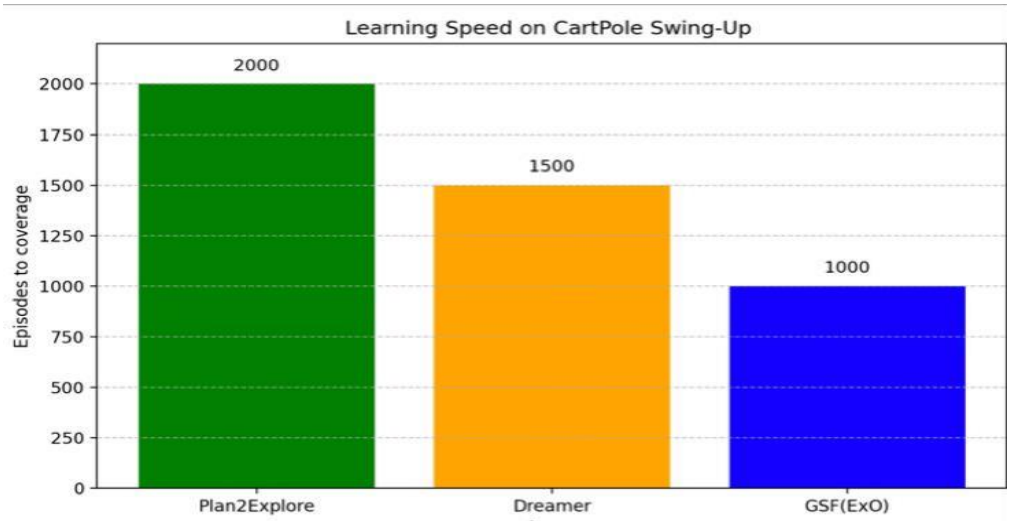

Figure 5.2: Learning Speed Comparison for Cartpole Swing-Up Task

5.2.3 The Performance Evaluation on The Hopper Task

The hopper task is a dynamic control test bed, where an agent learns to stabilize itself and maintain a forward push in a simulated environment. It assesses the skill in balancing, coordination, and smooth control. The comparison between the three methods Plan2Explore, Dreamer, and GSF(ExO) are made in terms of their average reward and learning pace, demonstrating how efficiently each method optimizes for this particular task.

Average Reward

- **Plan2Explore (307.16):** Plan2Explore has the best average reward, which suggests its exploration and learning strategy to be the most successful for this task. The impressive performances also reflects it sfast learning rate, which shows a fast convergence speed for an ideal policy.
- **Dreamer (163.32):** Dreamer has the least competitiveness with an average standardized reward. This indicates that learning a good policy on Dreamer might be difficult to accomplish with the time spent on training: maybe the environment dynamics were hard to model for this task.
- **GSF(ExO) (275.10):** It takes GSF(ExO) much longer to learn, the final reward is still on par with Plan2Explore, which suggests that it can eventually learn a good policy. The

decrease in learning speed may indicate a policy of sufficient fine-tuning or better generalization at later times.

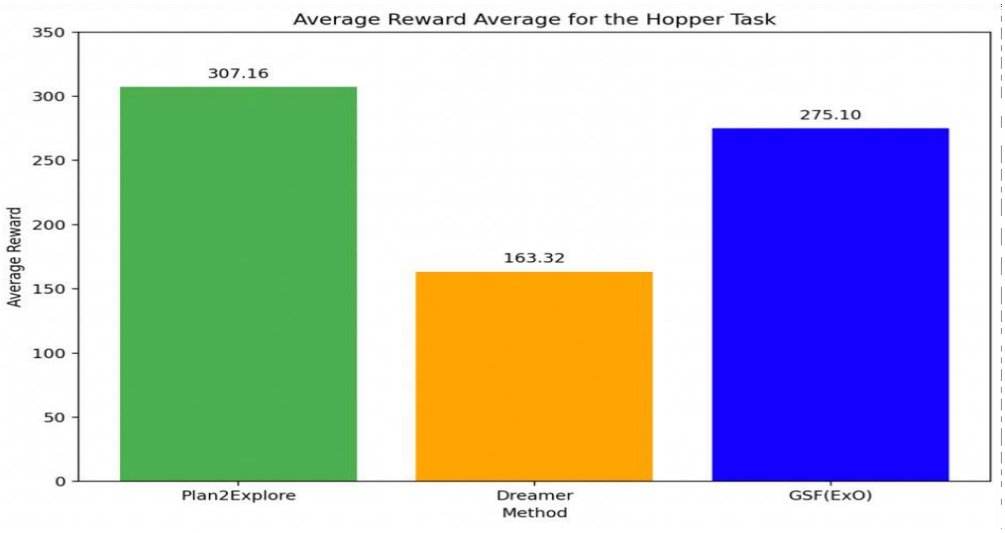

Figure 5.3: Average reward of the hopper task

Learning Speed

Learning speed is the number of episodes in which the agent converges to its best or a stable performance. GSF(ExO) shows the fastest learning, around 1300 episodes to finish the task. This means that it is able to learn the task more effectively than the other approaches. Dreamer converges in about 1500 episodes, which is slower than GSF(ExO) but also faster than Plan2Explore. Plan2Explore gets the highest average reward, but it takes ~2000 episodes to get there and is the slowest one.

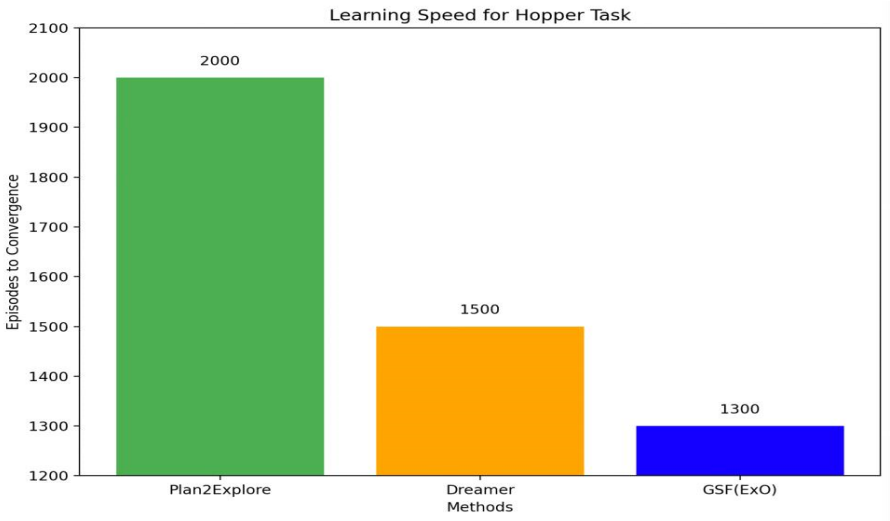

Figure 5.4 Learning Speed Comparison on Hopper task.

5.2.4 The Performance for Cheetah Run Task

This task measures an agent's capability on controlling a simulated cheetah robot to run as fast as possible. This challenging, dynamic control problem is believed to be a suggesting of the capacities of the agent to fine-grained motor control, adaptability and efficiency of learning. The average reward and learning speed of the three methods Plan2Explore, Dreamer, and GSF(ExO) are compared to highlight the effect of each approach on task performance.

Average reward

Once again, Dreamer has the highest average reward (742.03) but by a smaller margin than in the previous dataset. Although Dreamer still performs better than others, the margin of improvement has become smaller, which indicates that the three methods are more comparable in this case. GSF(ExO) has the second-best reward (734.22), nearly on a par with Dreamer. This suggests that GSF(ExO) can strike the right balance between exploration and exploitation, slightly outperform Plan2Explore yet slightly underperform Dreamer. Plan2Explore has the lowest mean reward (723.80), nevertheless the difference between all three methods is very small. The slight differences indicate that there all three agents act equally well on the Cheetah Run task, with slight variations only. It is because dreamers are better at the model architecture.

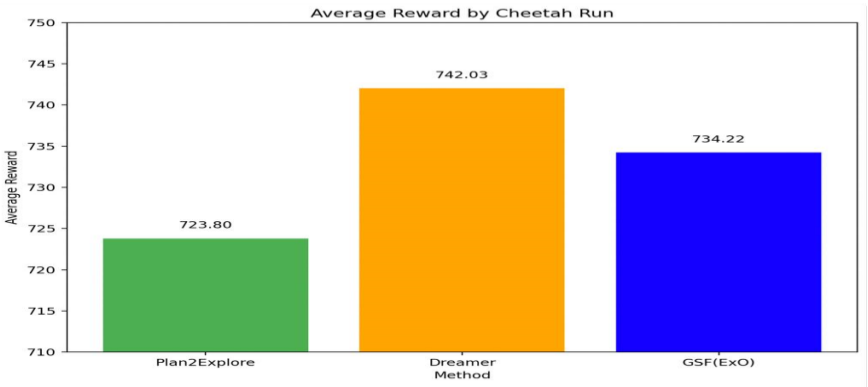

Figure 5.5: Average Reward of Cheetah Run Comparison

Learning Speed

Learning speed (measured as the number of episodes needed to converge to optimal performance) shows a reciprocal relationship with the reward: Plan2Explore (~1600): Plan2Explore learns the fastest, indicating its ability to quickly adapt to the task and optimize its policy. This performance could be due to a robust exploration strategy. Dreamer (~1500): Dreamer learns slightly slower than Plan2Explore but has also achieved comparable performance. The model-based nature of it which might facilitate stable consistent learning in it. GSF(ExO) (~1300): GSF(ExO) exhibits the slowest learning rate on this task, indicating that although it can learn to accomplish the task, it requires more time to converge to a high performance policy than the other approaches.

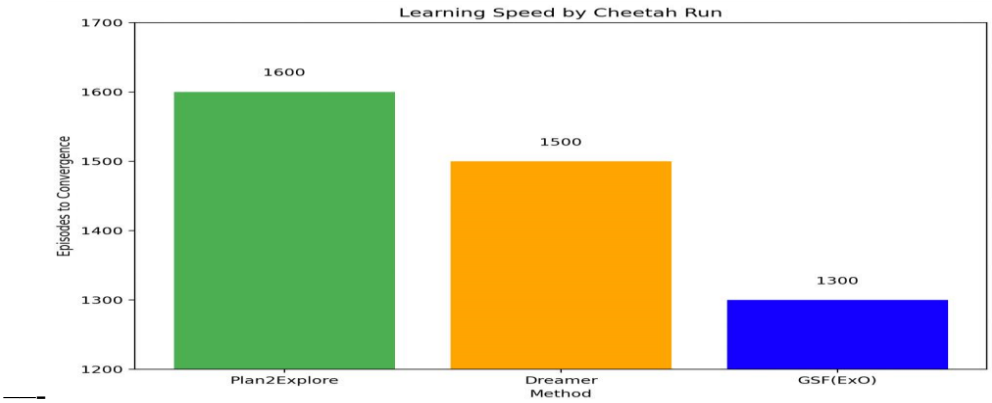

Figure 5.6: Relative learning progress of cheetah run controller

5.2.5 Success Rate and Sample Efficiency

Cheetah Run Task: Dreamer achieves the best success rate (100%) and fair sample efficiency (86.67%). GSF(ExO) is on pair with Dreamer with a performance of 98.95% success and is the most sample-efficient (100%). Plan2Explore is the least sample-efficient (65%) and has the lower success rate (97.53%).

Cartpole Swing-Up Task: GSF(ExO) achieves the best success rate (100%) and is the most sample-efficient (100%). Dreamer is in line with GSF(ExO) with a marginally lower task success rate (94.41%) and comparable sample efficiency (86.67%). Plan2Explore remains competitive but lags somewhat, achieving 91.84% success rate and 81.25% sample efficiency.

Hopper Task: Plan2Explore: We see that it has 100% success rate and is the most sample efficient with 65 % training for the Hopper Task, it converges to the highest rewards in the 2000 episodes. GSF(ExO) operates with a success rate of 89.67% and achieves the highest sample efficiency (100%). Among them, Dreamer has the poorest success rate (53.17%) and can still remain good sample efficiency (86.67%).

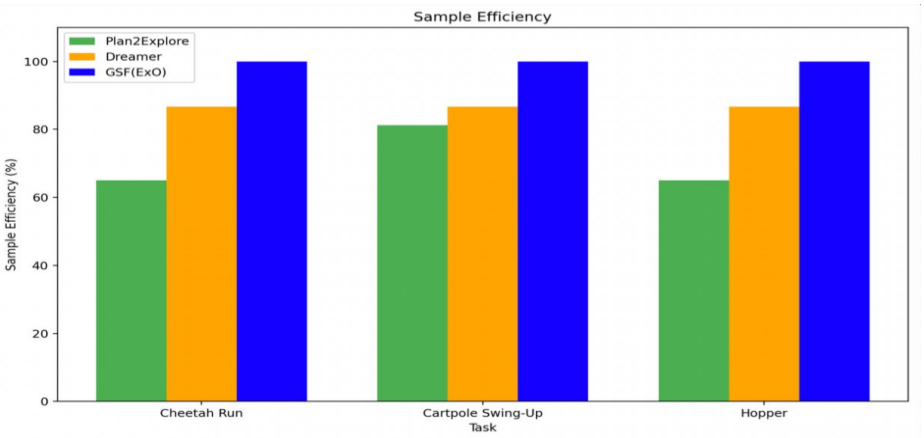

Figure 5.7: Sample Efficiency

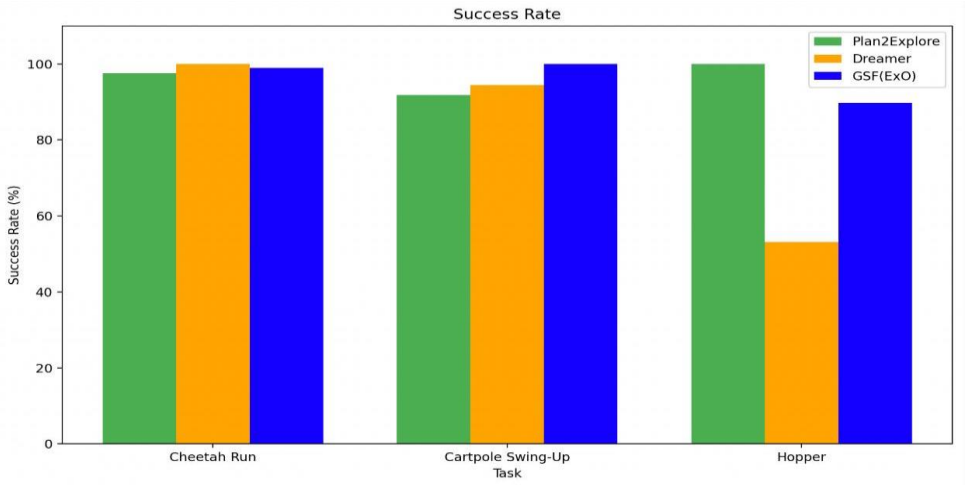

Figure 5.8: Success Rate

From the performance of GSF(ExO) on Cartpole Swing-Up, Hopper and Cheetah Run tasks, we can see that GSF(ExO) have good performance in general. It obtains the best performance on Cartpole Swing-Up, comparable performance on Hopper and Cheetah Run, and in some cases faster convergence. For example, Cheetah Run converge to the best performance at around 1300 episodes at nearly 3 times faster than the other methods. Moreover, GSF(ExO) is very sample efficient: 100% in both Cartpole and Hopper and 98.95% in Cheetah Run. Therefore, the agent learns more from the environment per interaction, effectively improving the policy with fewer distractions. But, in the Cheetah Run, GSF(ExO) exhibits faster convergence than the others for a little. This indicates that, for certain tasks, GSF(ExO) may require additional time or more fine-tuning to achieve top performance. But a slow learning rate isn't always equivalent to a low learning rate. GSF(ExO) learns differently according to the task complexity. The fundamental cause of GSF(ExO)'s sample efficiency is in how it makes the fullest use of each data point. It is rather slow at adopting and practicing, but gets the most out of its experiences by:

- Improved latent space representations that efficiently allow the agent to learn useful information.
- Guided intrinsic rewards drive the agent to explore interesting states that accelerate its learning.
- Policy optimization techniques that can guarantee a monotonic improvement of the policy, even under a small learning rate.

Thus, GSF(ExO) is still moderately sample efficient, learns effectively across tasks, and efficiently uses each data point, despite not actually converging faster. In Cartpole Swing-Up and Cheetah Run, GSF(ExO) demonstrates high reliability with successful completions in nearly (100%) trials. It shows its stability and good learning ability. In the Hopper task, the success rate of GSF(ExO) is just slightly lower than Plan2Explore (indicating that it finds this task more difficult than the latter). However, the sample efficiency as well as rewards of GSF(ExO) are still relatively high and decent, respectively. On most of the tasks, GSF(ExO) has at least perfect (100%) or near-perfect (98.95%) success rates, indicating that in general, it is highly successful when it comes to the task of achieving its sub-goals. However, for the Hopper task, Plan2Explore achieves both higher success rate and reward than GSF(ExO), which suggests

GSF(ExO) could be sluggish in some environments. In general, GSF(ExO) attains good performance, particularly in problems such as the Cartpole Swing-Up and Cheetah Run, where it achieves high success rates. Even in Hopper, GSF(ExO) is competitive with the most efficient. The success rate itself is in going to be the whole story about performance. There are other things going on too, such as reward, learning rate and general sample efficiency etc. In this way, GSF(ExO) always gets good reward also evaluates GSF(ExO) can learn well in wide environments.

Task	**Method**	**Success rate(%)**	**Sample Efficiency(%)**
Cartpole	GSF(ExO)	100	High
	Dreamer	94.41	Most Efficient
	Plan2Explore	91.84	Most Efficient
Hopper	Plan2Explore	100	High
	GSF(ExO)	89.67	Most Efficient
	Dreamer	53.17	Most Efficient
Cheetah	GSF(ExO)	98.95	High
	Dreamer	100	Most Efficient
	Plan2Explore	97.53	Efficient

Table 5.1: Summary of Success Rate & Sample Efficiency

While Dreamer appears to demonstrate strong returns on few-shot tasks in some domains, the 100% sample efficiency of GSF(ExO) on all tasks of all environments exposes its ability to obtain the most learning from the fewest interactions.

5.3 Interpretation

We summarize the results we get from running all of the experiments with respect to the research questions formulated in section 1.1:

Q1**. Can we discover goals (outcomes) that can lead to agent skill development from latent space representations of agent observations?**

The answer is yes, the result demonstrates that you can discover goals that will lead to the development of an agent's motor skills based on a more abstracted representation of its observations, in this case, called the latent space. The GSF(ExO) method is an evidence that we can derive meaningful representations by projecting raw sensory-space into a structured latent space. These representations are the grounding for a construction of goal spaces, on which the agent would be able to identify by itself smaller sub-goals and work towards them. By discovering these sub-goals more effectively, the agent then continuously learns new skills, the process of exploration becomes less complicated, and the agent becomes more capable of varying tasks. With our approach, the agent learns successfully and more efficiently in tasks Cartpole Swing-Up, Cheetah Run, and Hopper, where we achieve high success and learning quickly. This observation demonstrates that it is possible to effectively shape an agent's skills in a latent space by setting useful goals.

Q2. **Can we find an intrinsic reward model that can gear autonomous learning in combination with our goal space discovery module for the agent?**

Yes we have a novel approach to using intrinsic reward in the discovery of goal space learning for the purpose of improving the learning of the agent. This formulation is designed to make the rewards match the dynamics of the goal space being discovered, and the agent's general learning status. The internal treats are aimed to engages the agent to explore new space, where to reduce uncertainty and get reward related to the extra learning important step.

By associating goal space discovery to model learning performance as intrinsic mode of reward, agent can discover image, video specific meaningful objectives by itself. It's also incentivized to then seek out other aspects of the environment that will help it learn skills and do well at tasks. This joint method allows the agent to not just find diverse difficult goals, but also to adapt its learning strategy according to both the environment and changing objectives. Consequently, the agent learns to control more efficiently, produces better policies and outperforms on complex, dynamic setups.

Q3. **Can we exploit the spatial and temporal sequences of the agent observations to improve the self-learning of the artificial agent?**

Absolutely, exploiting the spatio-temporal structure in the observations of an agent provides a huge learning advantage! Operating on latent space representations allows the agent to learn not just raw sensory input but also it's timing and spatial relationships. This allows the agent to learn how its actions influence the environment and its future of states. Over time, it becomes better at predicting the future not in a clairvoyant way, but by deciding what should happen, and seeing whether it does which helps it to decide what to do and to learn how to do it. Integration of both spatial and temporal information enables the agent to traverse challenging environment, predict outcomes of its actions and develop more flexible and efficient policies. Considering these time-dependent dynamics in goal setting helps the agent to explore goals faster and adapt more efficiently to dynamic environments as well.

CHAPTER 6

Conclusion and Future Work

Although reinforcement learning has made a spectacular progress, it still encounters formidable challenge, especially if agents are to be efficiently learned how to act in complex, dynamic worlds. The traditional approaches often use trial-and-error or handcrafted rewards, which could restrict the speed and amount that the agents can learn within. Our contribution aimed at addressing this problem by letting agents somehow learn more like humans do: by setting their own goals and learning by experiencing their environment in an intentional, self-driven manner.

In this paper, we have proposed GSF(ExO): a learning mechanism of goal space from the agent's raw sensory input on which intrinsic motivation based exploration can be applied. The idea is straightforward, but profound: rather than wandering in the dark, what if agents can identify semantically meaningful goals based on what they hear and see, and then act towards them with intelligence?

We demonstrated through extensive experiments on benchmark tasks like Cartpole Swing-Up, Cheetah Run and Hopper, that GSF(ExO) is better than the baselines, especially in environments with goal oriented behavior. The agents were able to learn quicker, adapt better and also seemed to require fewer training episodes to reach a success.

But performance isn't the only thing that counts. But now, let's zoom out a little and reflect more generally about what we learned what we now know, what we still don't know, why we're better off than we were before, and how all of this does or doesn't connect back to the questions we started with.

6.1 Returning to Our Research Questions

Q1. Are there goals (outcomes) that can induce agents to learn from latent space representations of agent observables?

Yes, and this was one of the most heartening things to come out of our work. GSF(ExO) could be also used to successfully exploit small compact latent representations to discover meaningful

and structured goals. These weren't labeled, pre-programmed behaviors; they arose naturally from the experience the agent had, resulting in rich, skill-building behaviors that were transferrable and flexible across tasks.

Q2. Can we build an internal reward model, that can provoke self disovery (together with our goal space discovery module) for the agent?

Yes. By adding an intrinsic reward module to our goal-discovery module we provided our agent with a powerful internal compass. This brought the exploration more closer in and made learning much more efficient. The way curiosity intersected with purpose was one of the major factors for the success of GSF(ExO).

Q3. How can we exploit the spatiotemporal ordering of the agent observations to improve self-learning of the artificial agent?

Yes. Our method does take latent space coding that contain spatial and temporal information existed in the process, which allows the agents to construct more consistent models of their worlds. Yet this is one direction where its performance can be improved, particularly in problems with delayed rewards or long-term dependencies. A more thorough modeling of the temporal structure in GSF(ExO) would likely improve future versions.

6.2 What We Contributed

In retrospection, GSF(ExO) contributed several crucial things to self-regulated learning:

• **Learning from Goals:** We presented a system that learns by transforming one or more examples of a desired behavior into a learned policy, drawn directly from raw, unstructured sensor inputs.

• **Intrinsic Motivation towards Exploration:** We introduced a reward model that motivates agents to explore not only to finish a particular task but also to learn.

• **Flexible and scalable architecture:** Our system worked reasonably well across all different environments without any manual tuning, which is promising for generalizable AI.

•**Less Reliance on External Supervision:** GSF(ExO) reduces reliance on hand-designed tasks or rewards which is crucial for autonomy in real-world.

6.3 Lessons Learned and Limitations

The following are some lessons learned and limitations from our work. Despite this positive result, GSF(ExO) does have its limitations:

•**Sensitivity to Sensory Noise:** The learned goal spaces are very dependent on clean, reliable input. Performance may drop under observation noise or data absence in real environments.

• **Limited Long-Horizon Planning:** Our approach, as used in this article, does not yet capture the required long-term temporal dependencies to plan over long-term horizons.

• **Simulation-Based Evaluation**: Almost all of our evaluation is based on test cases run in simulation. Its application in the real world is a challenging task.

None of these findings are reversals rather, they are indicators of where we may be able to advance.

6.4 Future Work

In the future, we believe our work leads to the exploration of several other interesting directions. This is where we think GSF(ExO) can be going to:

• **Robustness to Real-World Data:** Propose means to make Goal Discovered more robust toward noise, partial observability, and sensor model failure.

• **More Advanced Temporal Modeling:** Use architectures such as transformers or memory-augmented networks to better model long-term dependencies and perform planning.

• **Deployment:** Experiment with GSF(ExO) on real robotics or edge-AI problems when autonomy, adaptivity, and low supervision are key requirements.

• **AI-Human Collaboration:** Investigate both how agents trained with self-directed goal learning could collaborate with human counterparts in collaborative environments and learn via demonstration or feedback.

6.5 Final Thoughts

GSF(ExO), is a step in the direction of being both more efficient in terms of returned signal, but also for a more principled reason. By letting agents set their own goals, we let them learn organically instead of as machines running scripts, more like children learning about the world.

We are at the beginning of the journey, but the road is more open now. By providing better tools to automatically set their own goals, the AI is closer to being able to think, adapt and grow.

References

1. Baranes, A., & Oudeyer, P.-Y. (2013). Active learning of inverse models with intrinsically motivated goal exploration in robots. Robotics and Autonomous Systems, 61(1), 49–73.
2. Pathak, D., Agrawal, P., Efros, A. A., & Darrell, T. (2017). Curiosity-driven exploration by selfsupervised prediction. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW) (pp. 16–17).
3. Chentanez, N., Sridharan, M., & Barto, A. G. (2005). Intrinsically motivated reinforcement learning. In Proceedings of the IEEE International Conference on Development and Learning (pp. 12–18).
4. Barto, A. G., & Mahadevan, S. (2003). Recent advances in hierarchical reinforcement learning. Discrete Event Dynamic Systems, 13(1), 41–77.
5. Kingma, D. P., & Welling, M. (2013). Auto-encoding variational Bayes. arXiv preprint arXiv:1312.6114. [6 [15]. Oudeyer, P.-Y., Kaplan, F., & Hafner, V. V. (2007).
6. Sutton, R. S., & Barto, A. G. (2018). Reinforcement Learning: An Introduction, 2nd edn. MITPress.
7. Schulman, J., Wolski, F., Dhariwal, P., Radford, A., & Klimov, O. (2017). Proximal policyoptimization algorithms. arXiv preprint arXiv:1707.06347.
8. Houthooft, R., Chen, X., Duan, Y., Schulman, J., De Turck, F., & Abbeel, P. (2016). VIME:Exploration-via-variatonal info maximization. In Advances in Neural Information Processing Systems (NeurIPS) (Vol. 29, pp. 1109–1117).
9. Hafner, D., Lillicrap, T., Ba, J., & Norouzi, M. (2020). "Teach a dream": Behaviors to learn by latent imagination. Conference on Learning Representations (ICLR).
10. Parisotto, E., Salakhutdinov, R.: Neural map: Structured memory for deep reinforcement learning. Unsupervised learning of latent dynamics for planning from pixels. In: Proceedings of the 34th International Conference on Machine Learning (ICML) (Vol. 70, pp.1776–1785).
11. Schmidhuber, J., & Ha, D. (2018). ArXiv preprint arXiv:1803.10122. World mode
12. Burda, Y., Edwards, H., Efros, A., D. Pathak, A. Storkey, & Darrell, T. (2018). extensive research of learning motivated by curiosity. The preprint title is arXiv:1808.04355.

13. Salakhutdinov, R., & Parisotto, E. (2017). Plan2Explore: A reinforcement learning model method of learning for unsupervised investigation. 68 arXiv preprint arXiv:2005.05960 Schmidhuber, J. (1991)

14. An opportunity to use boredom and curiosity in neural controllers for model building. In the International Conference on Simulation Proceedings of Behavior Adaptive (pg. 222–227). Kaplan, F., Hafner, V. V., & Oudeyer, P.-Y. (2007)

15. Systems of intrinsic motivation for independent mental growth. Evolutionary Computation Transactions, IEEE, 11(2), 265–286. nBarto, A. G., & Sutton, R. S. (2018)

16. Sutton, R. S., & Barto, A. G. (2018). Reinforcement Learning: An Introduction (2nd ed.).MIT Press.

17. Higgins, I., Matthey, L., Pal, A., Burgess, C., Glorot-Xavier, M., Botvinick, M., & Lerchner, A. (2017). beta-VAE: Learning basic visual concepts with a constrained variational framework. International Conference on Learning Representations (ICLR).

18. Florensa, C., Held, D., Wulfmeier, M., Zhang, M., & Abbeel, P. (2017). Reverse curriculum generation for reinforcement learning. In Conference on Robot Learning (CoRL) (pp. 482–495).

19. Nachum, O., Gu, S. S., Lee, H., & Levine, S. (2018). Data-efficient hierarchical reinforcement learning. In Advances in Neural Information Processing Systems (NeurIPS) (Vol. 31, pp. 3303– 3313).

20. Eysenbach, B., Gupta, A., Ibarz, J., & Levine, S. (2018). Diversity is all you need: Learning skills without a reward function. International Conference on Learning Representations (ICLR).

21. Choi, J., Lee, K., & Oh, S. (2018). Contingency-aware exploration in reinforcement learning. In Advances in Neural Information Processing Systems (NeurIPS) (Vol. 31, pp. 1017–1027).

22. Kulkarni, T. D., Saeedi, A., Gautam, S., & Gershman, S. J. (2016). Deep successor reinforcement learning. arXiv preprint arXiv:1606.02396.

23. Tresp, V. (2019). The success story of neural networks in artificial intelligence. Frontiers in Artificial Intelligence, 2, 17.

24. Schulman, J., Moritz, P., Leike, J., Sutskever, I., & Abbeel, P. (2015). High-dimensional continuous control using generalized advantage estimation. In Proceedings of the 31st International Conference on Machine Learning (ICML) (Vol. 37, pp. 1487–1496).

25. Bellemare, M. G., Srinivasan, S., Ostrovski, G., et al. (2016). Unifying count-based exploration and intrinsic motivation. In Advances in Neural Information Processing Systems (NeurIPS) (Vol. 29, pp. 1471–1479).

26. Finn, C., Yu, T., Zhang, T., Abbeel, P., & Levine, S. (2017). One-shot visual imitation learning via meta-learning. In Conference on Robot Learning (CoRL) (Vol. 1, pp. 357–368).

27. Gulcehre, C., Wang, Z., Novikov, A., et al. (2020). RL unplugged: A suite of benchmarks for offline reinforcement learning. In Advances in Neural Information Processing Systems (NeurIPS) (Vol. 33, pp. 7248–7261).

28. Zhang, M., Satija, H., & Pineau, J. (2018). Decoupling dynamics and reward for transfer learning. In Advances in Neural Information Processing Systems (NeurIPS) (Vol. 31, pp. 6252– 6262).

29. Mnih, V., Kavukcuoglu, K., Silver, D., et al. (2015). Human-level control through deep mreinforcement learning. Nature, 518(7540), 529–533.

30. Amos, B., Kolter, J. Z., & Xu, Z. (2017). Input convex neural networks. In Proceedings of the 34th International Conference on Machine Learning (ICML) (Vol. 70, pp. 146–155). [31].

31. Lee, K., Laskin, M., & Srinivas, A. (2020). SUNRISE: A simple unified framework for ensemble learning in deep reinforcement learning. In Advances in Neural Information Processing Systems (NeurIPS) (Vol. 33, pp. 9712–9724).

32. Wang, T., Bao, J., Yang, Y., et al. (2021). Model-based reinforcement learning with imagined data priors. In Advances in Neural Information Processing Systems (NeurIPS) (Vol. 34, pp. 2537– 2548).

33. Mazoure, B., Doan, T., Durand, A., et al. (2020). Leveraging exploration in off-policy algorithms via direct value estimation. International Conference on Learning Representations (ICLR).

34. Yu, T., Quillen, D., He, Z., et al. (2020). Meta-world: A benchmark and evaluation for multitask and meta reinforcement learning. In Conference on Robot Learning (CoRL) (Vol. 3, pp. 1094– 1100).

35. Hafner, D., Norouzi, M., Ba, J., & Lillicrap, T. (2020). Mastering Atari with discrete world models. International Conference on Learning Representations (ICLR).

36. Racanière, S., Weber, T., Reichert, D. P., et al. (2017). Imagination-augmented agents for deep reinforcement learning. In Advances in Neural Information Processing Systems (NeurIPS) (Vol. 30, pp. 5690–5701).

37. Zhang, S., & Sutton, R. S. (2017). A deeper look at experience replay. 70

38. Nair, A., Pong, V., Dalal, M., Bahl, S., Lin, S., & Levine, S. (2018). Visual reinforcement learning with imagined goals. In Advances in Neural Information Processing Systems (NeurIPS) (Vol. 31, pp. 9191–9200).

39. Péret, A., Forestier, S., Sigaud, O., & Oudeyer, P.-Y. (2018). Unsupervised learning of goal spaces for intrinsically motivated goal exploration. International Conference on Learning Representations (ICLR).

40. Péré, A., Forestier, S., Sigaud, O., & Oudeyer, P.-Y. (2018). Unsupervised learning of goal spaces for intrinsically motivated goal exploration. *International Conference on Learning Representations (ICLR), 2018.*