BACHELOR'S DEGREE THESIS

Degree in Data Science and Engineering

Title: Unsupervised Skill Discovery and Learning from Pixels

Author: Roger Creus Castanyer

Advisor: Xavier Giró-i-Nieto, Juan José Nieto Salas

Department: Signal Theory and Communications Department

Month and year: June 2021

UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH

Facultat d'Informàtica de Barcelona
Facultat de Matemàtiques i Estadística
Escola Tècnica Superior d'Enginyeria de Telecomunicació de

Acknowledgments

I want to thank my two supervisors Xavier Giró-i-Nieto and Juan José Nieto Salas for the unconditional help they have brought me. They have always proposed and debated ideas with me in order to develop the flow of this project. On the one hand, Xavier Giró has enabled me to successfully link my work with the research domain, for what I am very glad. On the other hand, Juan José Nieto has taught me lots of domain-specific concepts and has supported me technically all along the project. I also want to thank Víctor Campos for reviewing our related publications and providing valuable guidance and feedback.

I want to show my appreciation to my beloved ones. I am very grateful for being surrounded by supportive family and friends.

This work was partially supported by the Postgraduate on Artificial Intelligence with Deep Learning of UPC School, and the Spanish Ministry of Economy and Competitivity under contract TEC2016-75976-R. We gratefully acknowledge the support of NVIDIA Corporation with the donation of GPUs used in this work.

Abstract

In an attempt to enable agents carrying Artificial Intelligence (AI) components to solve quotidian real world problems, the behaviours that define the intelligence of the agents must adapt to the nature of the real world environment. In the Deep Learning (DL) field it has been shown that deep Neural Networks (NN) are outstanding models for learning in a supervised way. In this way, NNs have achieved state-of-the-art performance in many supervised tasks, even outperforming humans (i.e. image, object and voice recognition). However, for many tasks it is not possible to label the real world environment due to its complexity. Hence, agents carrying NNs trained in a supervised way will eventually underfit the real world environment. Self-Supervised Learning (SSL) and Reinforcement Learning (RL) are approaches that aim to enable AI models to obtain knowledge without the assumption of extrinsic annotations as in Supervised Learning (SL). Actually, SSL and RL approaches are much closer to how the humans learn from the real world environment. For instance, when we are babies, we do not learn how to walk by only watching examples of people walk. We learn to walk by a combination of exploration of behaviours plus the interaction with the real world. Within this example, an interaction with the real world could be pain in the baby's body for having fallen after misplacing its legs when trying to stand up. Also, babies eventually prefer walking rather than crawling because they learn to distinguish the value of the two behaviours: walking is more valuable than crawling as it allows one to reach a navigation goal, or perform auxiliary actions with a smaller effort and a higher precision. Along this example, the majority of the fundamental human knowledge, and specially the one that implies interaction with the real world environment is self-obtained by means of action-value estimation. This is all thanks to the brain processing capabilities in combination of the human-world reward interactions. Arguably, the aforementioned statements about the human learning processes considerably intersect with the principles of SSL and RL.

Among all the scenarios where the AI capabilities can provide a valuable service to humans, robotics might be the most conditioned one by the real world environment. In this field of embodied AI, the most sophisticated robots nowadays integrate natural language processing, speech generation, face recognition, motor control, and many other capabilities. However, if we drop one of these most sophisticated robots in our homes and tell them to perform any human quotidian action (e.g, look for the lost TV remote control, put the pizza in the oven, close the bathroom door) they will fail drastically. The robots will fail because there is an implicit difficulty in the accomplishment of generalization across multiple tasks, no matter how simple the tasks are. That is because for efficiently solving a set of downstream tasks, embodied AI agents at least need to *(i)* Perceive the environment: for instance through vision sensors as RGB or Depth observations; *(ii)* Act: navigate and interact with their environment to accomplish goals; and *(iii)* Reason: consider and plan for the long-term consequences of their actions.

This work focuses on the self-acquirement of the fundamental task-agnostic knowledge available within an environment. The aim is to discover and learn baseline representations and behaviours that can later be useful for solving embodied visual navigation downstream tasks. In this way, the difficulty of the embodied multitask learning problem is simplified by first obtaining an intelligent perception of the environment and its dynamics. Following the previously mentioned examples of human quotidian tasks (i.e. close the bathroom door or put a pizza in the oven), this work motivates that first knowing how to recognize the bathroom, what makes it different from the kitchen, and learning to navigate between both will facilitate both downstream tasks.

Specifically, the presented approach extends the idea of the Explore, Discover and Learn (EDL) paradigm

to the pixel domain. This way, this work is centered in the representations and behaviours that can be learnt by an agent that only integrates an image capture sensor. Both the agents and the environment that is used in this work run over the Habitat AI simulator, which is developed by Facebook AI, and renders 3D fotorealistic views of the insides of apartments.

Resum

Per tal que els agents que inclouen components d'Intel·ligència Artificial (IA) puguin resoldre problemes quotidians del món real, els comportaments que defineixen la intel·ligència d'aquests agents s'han d'adaptar a la seva naturalesa. En el camp de l'aprenentatge profund, *deep learning (DL)*, s'ha demostrat que les xarxes neuronals profundes (NN) són models excel·lents per aprendre de manera supervisada. És així com les NN han aconseguit un rendiment excepcional en una gran varietat de tasques supervisades, fins i tot superant els humans (ex. reconeixement d'imatges, objectes i veu). No obstant això, no és possible etiquetar l'entorn del món real per a moltes d'altres tasques a causa de la seva complexitat. Per tant, en molts casos els agents que inclouen xarxes entrenades de manera supervisada no poden captar la complexitat del món real. L'aprenentatge autocontrolat, *self-supervised learning (SSL)*, i l'aprenentatge de reforç, *reinforcement learning* (RL), són enfocaments que tenen com a objectiu que els models d'IA puguin obtenir coneixement sense l'ús d'anotacions extrínseques, com a l'aprenentatge supervisat (SL). En realitat, els enfocaments SSL i RL són molt similars als dels aprenentatges que fan els humans del seu entorn, el món real. Per exemple, quan som nadons, no aprenem a caminar mirant només exemples de gent que camina. Aprenem a caminar mitjançant la combinació de l'exploració de comportaments i la interacció amb el món real. Seguint amb aquest exemple, una interacció amb el món real podria ser el dolor que pot sentir el nadó si, després de posicionar malament les cames en intentar posar-se dret, cau a terra. A més, els nadons eventualment prefereixen caminar en lloc d'arrossegar-se, perquè aprenen a distingir el valor dels dos comportaments: caminar és més valuós que arrossegar-se, ja que permet assolir un objectiu de navegació o realitzar accions auxiliars amb un esforç menor i una precisió més alta. Juntament amb aquest exemple, la majoria del coneixement humà fonamental, i especialment el que implica la interacció amb l'entorn del món real, s'obté autònomament per mitjà d'exploració i estimació del valor dels nostres comportaments. Tot això passa gràcies a les capacitats de processament del cervell en combinació de les interaccions de recompensa amb l'entorn. Es podria dir que les afirmacions esmentades sobre els processos d'aprenentatge humans es creuen considerablement amb els principis de SSL i RL. De tots els escenaris en què les capacitats d'IA poden proporcionar un servei valuós als humans, la robòtica pot ser la més condicionada per l'entorn del món real. En aquest camp de la IA encarnada, els robots més sofisticats integren avui dia processament del llenguatge natural, generació de veu, reconeixement facial, control sensorial i moltes altres capacitats. Tanmateix, si despleguem un d'aquests robots més sofisticats a casa nostra i li diem que faci qualsevol acció quotidiana humana (per exemple, buscar el comandament del televisor, introduir la pizza al forn, tancar la porta del bany) fracassarà dràsticament. El robot fallarà perquè hi ha una dificultat implícita en la generalització entre vàries tasques, per molt senzilles que puguin semblar. Això es deu al fet que, per resoldre de manera eficient un conjunt de tasques, els agents d'AI com a mínim han de *(i)* Percebre l'entorn: per exemple, mitjançant sensors de visió com RGB o observacions de profunditat; *(ii)* Actuar: navegar i interactuar amb el seu entorn per assolir els objectius; i *(iii)* Raonar: considerar i planificar les conseqüències a llarg termini de les seves accions. Aquest treball se centra en l'adquisició del coneixement fonamental agnòstic de tasques disponibles en un entorn. L'objectiu és descobrir i aprendre representacions i comportaments generalistes que puguin ser útils més endavant per resoldre tasques de navegació visual.

La dificultat del problema d'aprenentatge multi-tasca se simplifica si primer s'obté una percepció intel·ligent de l'entorn i de les seves dinàmiques. Seguint els exemples esmentats anteriorment de tasques quotidianes humanes (és a dir, tancar la porta del bany o posar una pizza al forn), aquest treball motiva que primer s'ha de saber reconèixer el bany (què el fa diferent de la cuina?) i després aprendre a navegar entre aquests dos espais facilitarà les dues tasques. En concret, l'enfocament presentat amplia la idea de la tècnica *Explore, Discover and Learn (EDL)* al domini dels píxels. D'aquesta manera, aquest treball se centra en les representacions i comportaments que poden ser descoberts i apresos per un agent sense supervisió i que només inclou un sensor de captura d'imatges. Tant els agents com l'entorn que s'utilitzen en aquest treball funcionen amb el simulador d'Habitat AI, desenvolupat per Facebook AI, que proporciona vistes fotorealistes en 3D de l'interior d'apartaments.

Resumen

Para que los agentes que incluyen componentes de Inteligencia Artificial (IA) puedan resolver problemas cotidianos del mundo real, los comportamientos que definen la inteligencia de estos agentes deben adaptarse a su naturaleza. En el campo del aprendizaje profundo, textit deep learning (DL), se ha demostrado que las redes neuronales profundas (NN) son modelos excelentes para aprender de manera supervisada. Es así como las NN han conseguido un rendimiento excepcional en una gran variedad de tareas supervisadas, incluso superando los humanos (ej. Reconocimiento de imágenes, objetos y voz). Sin embargo, no es posible etiquetar el entorno del mundo real para muchas otras tareas debido a su complejidad. Por lo tanto, en muchos casos los agentes que incluyen redes entrenadas de forma supervisada no pueden captar la complejidad del mundo real. El aprendizaje autocontrolado, textit self-supervised learning (SSL), y el aprendizaje de refuerzo, textit Reinforcement learning (RL), son enfoques que tienen como objetivo que los modelos de IA puedan obtener conocimiento sin el uso de anotaciones extrínsecas, como el aprendizaje supervisado (SL). En realidad, los enfoques SSL y RL son muy similares a los de los aprendizajes que hacen los humanos de su entorno, el mundo real. Por ejemplo, cuando somos bebés, no aprendemos a caminar mirando sólo ejemplos de gente que camina. Aprendemos a caminar mediante la combinación de la exploración de comportamientos y la interacción con el mundo real. Siguiendo con este ejemplo, una interacción con el mundo real podría ser el dolor que puede sentir el bebé si, después de posicionar mal las piernas al intentar ponerse de pie, cae al suelo. Además, los bebés eventualmente prefieren caminar en lugar de arrastrarse, porque aprenden a distinguir el valor de los dos comportamientos: caminar es más valioso que arrastrarse, ya que permite alcanzar un objetivo de navegación o realizar acciones auxiliares con un esfuerzo menor y una precisión más alta. Junto con este ejemplo, la mayoría del conocimiento humano fundamental, y especialmente lo que implica la interacción con el entorno del mundo real, se obtiene autónomamente por medio de exploración y estimación del valor de nuestros comportamientos. Todo esto ocurre gracias a las capacidades de procesamiento del cerebro en combinación de las interacciones de recompensa con el entorno. Se podría decir que las afirmaciones mencionadas sobre los procesos de aprendizaje humanos se cruzan considerablemente con los principios de SSL y RL. De todos los escenarios en que las capacidades de IA pueden proporcionar un servicio valioso a los humanos, la robótica puede ser la más condicionada por el entorno del mundo real. En este campo de la IA encarnada, los robots más sofisticados integran hoy en día procesamiento del lenguaje natural, generación de voz, reconocimiento facial, control sensorial y muchas otras capacidades. Sin embargo, si desplegamos uno de estos robots más sofisticados en nuestra casa y le decimos que haga cualquier acción cotidiana humana (por ejemplo, buscar el mando del televisor, introducir la pizza en el horno, cerrar la puerta del baño) fracasará drásticamente. El robot fallará porque

hay una dificultad implícita en la generalización entre varias tareas, por muy sencillas que puedan parecer. Esto se debe a que, para resolver de manera eficiente un conjunto de tareas, los agentes de AI como mínimo deben it (i) Percibir el entorno: por ejemplo, mediante sensores de visión como RGB u observaciones de profundidad; It (ii) Actuar: navegar e interactuar con su entorno para alcanzar los objetivos; y it (iii) Razonar: considerar y planificar las consecuencias a largo plazo de sus acciones. Este trabajo se centra en la adquisición del conocimiento fundamental agnóstico de tareas disponibles en un entorno. El objetivo es descubrir y aprender representaciones y comportamientos generalistas que puedan ser útiles más adelante para resolver tareas de navegación visual. La dificultad del problema de aprendizaje multi-tarea se simplifica si primero se obtiene una percepción inteligente del entorno y de sus dinámicas. Siguiendo los ejemplos mencionados anteriormente de tareas cotidianas humanas (es decir, cerrar la puerta del baño o poner una pizza en el horno), este trabajo motiva que primero hay que saber reconocer el baño (que lo hace diferente de la cocina?) y luego aprender a navegar entre estos dos espacios facilitará las dos tareas. En concreto, el enfoque presentado amplía la idea de la técnica textit Explore, Discover and Learn (EDL) al dominio de los píxeles. De este modo, este trabajo se centra en las representaciones y comportamientos que pueden ser descubiertos y aprendidos por un agente sin supervisión y que sólo incluye un sensor de captura de imágenes. Tanto los agentes como el entorno que se utilizan en este trabajo funcionan con el simulador de Habitat AI, desarrollado por Facebook AI, que proporciona vistas fotorrealistas en 3D del interior de apartamentos.

Contents

1 Introduction 7

2 Background 9

2.1 Introduction to Artificial Intelligence . . . 9

2.1.1 A taxonomy of Machine Learning . . . 9

2.2 Preliminaries of Reinforcement Learning . . . 11

2.3 Challenges . . . 12

3 Related Work 15

4 Goals and Problem Formalization 17

5 Proposed Solution 20

5.1 Exploration . . . 20

5.2 Skill Discovery . . . 20

5.2.1 Contrastive . . . 22

5.2.2 Reconstruction . . . 23

5.3 Skill Learning . . . 25

6 Results 27

6.1 Skill Discovery . . . 27

6.1.1 RGB sensor . . . 28

6.1.2 Depth sensor . . . 30

6.1.3 Instance segmentation sensor . . . 31

6.1.4 RGB + Depth + Instance Segmentation . . . 33

6.1.5 Generalization among different apartments . . . 34

6.2 Skill Learning . . . 36

6.3 Image Navigation . . . 37

7 Conclusions 41

1. Introduction

In this work we try to develop autonomous and intelligent robots (agents) that navigate effectively in a 3D realistic environment. For that, we consider agents that are fed with visual and depth data, which can be captured with low cost sensors available for robots.. Also, we aim to avoid human supervision for making the agents obtain generalist knowledge from the real world environment. We dive into the field of embodied artificial intelligence (AI) for deploying the agents with autonomous behaviours, and we consider the latter as the basis of assistive robotics capable of empowering people with reduced mobility at home, so that they can autonomously fulfill their daily living activities. Overall, this thesis motivates the hypothesis that an environment is full of characteristics that if processed into generalist knowledge it would be valuable for solving any downstream task in the environment. With all this, we expect truly autonomous embodied AI agents to function in complex environments and without needing expert human supervision.

We adopt the paradigm of Explore, Discover and Learn (EDL) [1] for allowing the agents to obtain a generalist and rich perception of their surroundings in an unsupervised manner. Concretely, we aim to train agents that get to understand the rich visual information in a real world context with the goal to develop knowledge representations and behaviours that generalize. EDL defines an information-theoretic objective and makes some assumptions in its formulation that do not scale to the pixel space. In this way, we are inspired from EDL to achieve our goals, and at the same time, we extend EDL to work for egocentric visual data from a 3D virtual world, which has not been tested yet. This work has been developed in parallel with the excellent Master's Thesis by Juan José Nieto Salas named *Discovery and Learning of Navigation Goals from Pixels in Minecraft*. Compared to the latter, we extend EDL to work in the Habitat AI phototorealistic environment, we leverage different sources of visual data (depth and semantic segmentation), we do not make use of human experts to train the agents, we use very different algorithms for defining the behaviours of the agents, and we provide a comparative analysis between our approach and a baseline to tackle image goal-oriented embodied navigation.

We deploy our implementation of EDL in the Habitat AI simulator [2]. Habitat is a high-performance 3D simulator that enables experimentation in embodied AI. It provides configurable agents, multiple sensors (i.e. GPS, compass, RGB, depth-channel and instance segmentation annotations) and high-performance rendering of photorealisitic home interiors. Moreover, Habitat contains several set-ups for solving multiple downstream tasks (i.e. visual and GPS-guided navigation, instruction following, embodied question answering).

In this work we focus on embodied visual navigation. Generally, embodied visual navigation relates to downstream tasks that are performed within a 3D environment, where there is a need to perform effective navigation and the instructions to the agents consist of only visual data. In this way, the Habitat AI environment proposes a challenging real-world-related context in terms of visual and dynamics complexity. In experimentation, we make use of different types of image data available in the Habitat AI environment, mainly RGB observations, depth channel information and instance segmentation annotations. With this, we assess the quality of the discovered representations that our agents adopt and find that we manage to obtain a valuable perception of the environment in terms of visual resemblance and spatial relations. Also, we train the agents in a self-discovered task to provide a framework for obtaining generalist behaviours in an environment. Finally, we propose an approach for transferring the knowledge representations that we obtain in an unsupervised way to agents that tackle image goal-conditioned navigation (ImageNav). ImageNav is an episodic task where at each episode the agents are set to an initial location and also set a goal location. However, the representation of this goal consists of only an image, and the objective is to

obtain agents that perform effective 3D navigation to reach the navigation goals specified by the images. With this set-up, we demonstrate that the generalist knowledge of the agents that we develop enables them to overcome the baseline approach for tackling the ImageNav task. This work has been presented in a peer-reviewed CVPR 2021 workshop on embodied AI.

2. Background

2.1 Introduction to Artificial Intelligence

Artificial Intelligence (AI) plays a key role in the world we live in. AI enables solutions as diverse as machine translation, self-driving cars, voice assistants, character and handwriting recognition, ad targeting, product recommendations, music recognition, and facial recognition. Moreover, AI overcomes us as humans in many tasks.

AI is about making machines mimic intelligent behaviours. For that, Machine Learning (ML) sets the foundations of AI. ML works over the most valuable and key source of knowledge that exists: data. In ML, machines take in data and learn patterns that would be difficult for humans to learn. ML is valuable in the sense that the data processing (or data classification, segmentation, representation) capabilities go far beyond than what humans can achieve. Some of the most applied ML models are: SVMs [3], Classification Trees [4] or Logistic Regression [5] for classification (the models predict the categorical class membership of the input data); Lasso/Ridge Regression [6, 7] and Regression Decision Trees for regression [8] (the models predict a numerical value usually in the continuous domain); K-means [9] and Agglomerative Clustering [10] for clustering (the models learn representations of the input data that provide a meaningful segmentation of the whole data set); and PCA [11] and SVD [12] for dimensionality reduction (the models learn simple representations of the data that preserve the complexity and variability of the entire data set).

2.1.1 A taxonomy of Machine Learning

In the wide field of ML there exists a distinguishment among several methodologies depending on the way the machines process the data into valuable knowledge:

- Supervised Learning (SL): It is based on input-output pairs of data. The machines infer a function that maps the input data to the output label. It demands a considerably large amount of input (training) examples. Otherwise, the inferred function's complexity can overfit the training examples not providing generalization.
- Unsupervised Learning (UL): It learns patterns from untagged data. Some well-known capabilities of UL is inferring probability densities, as in principal component and clustering analysis.
- Self-Supervised Learning (SSL): It falls between SL and UL. It is about self-crafting a proxy supervised objective within an untagged amount of data. In this way, machines can learn valuable representations from an untagged data set in a supervised way. A popular successful application of SSL is found in the training process of massive language models. Starting from a huge set of unlabelled text data and with the manipulation of this data (i.e. masking a random word in each sentence to the machine) a supervised objective is defined: predict the missing word. For which objective we have available ground truth as in SL (we just transformed a part of the data into labels).
- Representation Learning: It approaches the form in which the inputs are encoded before being fed into an AI component. In ML, a very important part of the capabilities (i.e. accuracy) that a model provides is due to the appropriate representation of the inputs. However, in DL representation learning

relates the embedded process in which NNs learn a mapping between inputs and representations. The representation space (in this work also categorized as embedding/latent space) aims to provide a simplification of the complex input space by relying in a low-dimensional manifold that preserves the input space variability.

- Reinforcement Learning (RL): RL changes the paradigm and formalism of the other approaches of ML. RL is about mapping situations to actions in order to maximize a reward signal within an environment. RL provides a framework for enabling embodied AI components (agents) to learn by interaction with an environment.

It makes sense to relate the aforementioned taxonomy of ML to "traditional AI". By traditional we understand the AI-enabled systems that are embedded in software applications/programs and depend on digital sources of data. For instance, Natural Language Processing (NLP) (i.e. sentiment analysis, language generation, machine translation), or image-based processing (i.e. object recognition, image captioning, trajectory-prediction) rely on massive digital sources of data (i.e. Sentiment140 or 20newsgroups for NLP and ImageNet [13] or COCO [14] for images). However, embodied AI also benefits from the traditional AI capabilities. By embodied AI we relate to AI components that are embedded into real world bodies. In this way, embodied AI can be the result of the deployment of sophisticated language or computer vision models into real world bodies. Actually, the most advanced robots nowadays are capable of recognising faces and emotions, sensing obstacles and even handling conversations (text and speech recognition and generation). However, all these capabilities are pretrained digitally and deployed into the robots. In fact, when the ML models that allow the aforementioned capabilities are digitally trained, they rely on shuffled, randomized packages of data, and this process does not map to the way in which humans learn. Humans learn by sequentially perceiving, moving and interacting with the environment. Mimicking the human learning processes in the real world is essential and becomes the major challenge of embodied AI. Intelligent robots could benefit from capabilities as incredible as effective navigation (i.e. self-driven cars), quotidian multitask handling or embodied question answering (i.e. service robotics, robots in healthcare).

A characteristic of embodied AI is that is ruled by egocentric perception. All the interaction of embodied AI agents come from a first-person view of the environment so all the objects are encoded with respect to the agents. In this sense, embodied AI finds its similarity with the human condition. For performing effective learning processes in the real world, embodied AI agents at least need to *(i)* Perceive the environment: for instance through vision sensors as RGB or Depth observations; *(ii)* Act: navigate and interact with their environment to accomplish goals; and *(iii)* Reason: consider and plan for the long-term consequences of their actions. In this way, embodied AI agents implicitly deal with interactions within an environment. Hence, it makes sense to relate RL to embodied AI. RL is the main methodology for providing embodied AI agents the intelligent capabilities of *(i)* mapping the agent observations to actions in the environment; *(ii)* assessing the value of their behaviours to accomplish goals; and *(iii)* adapting/tuning their behaviours/policies to optimize their performance in a given task.

In many cases, the real world is a too expensive environment for massively experimenting with embodied AI agents. In this way, there exist many simulated (virtual) graphical environments nowadays that are adapted to the training processes of RL [15, 16, 2]. Each of these serves a specific pipeline to solve an embodied AI related task (i.e. visual navigation, robotic manipulation, policy optimization, multi-agent learning). The standardised pipeline for obtaining real world intelligent behaviours nowadays consists of deploying virtual embodied AI agents in the aforementioned simulators and then transferring the obtained behaviours into real world bodies.

In the following, RL is characterised in order to specify the context of this work.

2.2 Preliminaries of Reinforcement Learning

In RL the characterization of an environment comes with the definition of the sets of states $s \in \mathcal{S}$ and actions $a \in \mathcal{A}$, and possibly the reward signal $R(s,a) \in \mathcal{R} \subset \mathcal{R}$ and the distribution of transition probabilities $p(s', r|s, a)$. The fact that the reward signal and the transition probabilities are known or not makes the difference between model-based and model-free RL. Learners in model-based RL resort to planning, which involves predicting the future states that will be consequences the of taken actions. In contrast, learners in model-free RL learn the best policies explicitly by trial-and-error. Another core element of the RL set-up is the policy $\pi(a|s)$ which expresses the agents behaving at a given time by defining a mapping between states and actions. With this framework, at each step the RL agents visit a state, take an action in the current state and receive guidance with the reward outcome. Ultimately, the goal of the RL agents is to maximize the total sum of rewards over the long run, namely the return:

$$G_t = R_t + \gamma R_{t+1} + \gamma^2 R_{t+2} + \gamma^3 R_{t+3} + ...$$

Where $\gamma \in [0, 1)$ is a discount factor that makes the agent put the priority in the immediate rewards. Along the definition of this goal, an interesting observation arises: The only way to know how valuable is taking an action in a given state is by exploring so. In the most challenging cases, the value of an action in a given state might not be defined by the immediate reward but by the long-term subsequent rewards that will come from the transitioned state. In this way, the RL core tasks are *(i)* exploration of the possible state-action pairs in an environment together with the value of these (which is characterised by $R(s,a)$, which at the same time is usually stochastic); and *(ii)* exploitation of behaviours that visit and take the most valuable state-action pairs.

For formally describing an environment in RL, we define Markov Decision Processes (MDP) that frame the underlying rules and dynamics of the environment. MDPs work over the Markov assumption: any action affects the immediate reward and the next state. What this assumption means is that any state visitation is only consequence of the last action taken. MDPs are defined by a tuple: $\mathcal{M} = (\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma)$ where $\mathcal{S}$ is the set of states $s \in \mathcal{S}$, $\mathcal{A}$ is the set of actions $a \in \mathcal{A}$, $\mathcal{R}$ is a reward function taking values $R(s,a) = \mathbb{E}(R_{t+1}|S_t = s, A_t = a)$, $\mathcal{P}$ is the transition probability distribution that characterizes the environment dynamics $p(s'|s,a) = Pr(S_{t+1} = s'|S_t = s, A_t = a)$ and γ is the aforementioned discount factor. With this set-up, if one wants to maximize the expected total reward G_t the goal to be accomplished is to learn the policy that obtains the highest expected total reward $\pi^*(a|s) = argmax_\pi \mathbb{E}_\pi(G_t)$.

With the goal of the maximisation of the expected total reward in mind, two key concepts arise. In the following it is defined the way in which the agent learns to estimate the value of its behaviours: either the value of visiting a state and the value of taking a specific action in that state.

The value of the state s is defined as the expected cumulative reward that will be obtained following the current policy:

$$v_\pi(s) = \mathbb{E}_\pi(G_t|S_t = s)$$

where $v_\pi(s)$ is the *state-value function*.

Then, the value of taking an action a in a given state s is the expected total reward that will follow the decision of taking a in s and following the current policy:

$$q_\pi(s,a) = \mathbb{E}_\pi(G_t|S_t = s, A_t = a)$$

Now, taking into account that $\mathbb{E}(G_t) = \mathbb{E}(R_t + \gamma\mathbb{E}(G_{t+1}))$ is recursive, two very important equations of RL arise:

$$v_\pi(s) = \mathbb{E}_\pi(R_t + \gamma v_\pi(S_{t+1})|S_t = s)$$

which is the Bellman expectation equation for the state-value function. And we have another recursive expression for the action-value function:

$$q_\pi(s, a) = \sum_{r,s'} p(s', r|s, a)[r + \gamma \sum_{a'} \pi(a'|s')q_\pi(s', a')]$$

These two equations provide the foundation of all the RL algorithms, since they provide a framework for deriving different methodologies for estimating the values of the states and the actions. Some examples of RL techniques are *(i)* Monte-Carlo (MC) methods, that use sampling for estimating the value functions; *(ii)* dynamic programming (DP) methods, that derive an update formula which uses the current estimate of the functions for the estimation in the next time step; *(iii)* temporal-difference (TD) methods, that combine *(i)* and *(ii)* for deriving update formulas. Intuitively, these methods iteratively update the state and action values with the reward distribution that the agents observe when visiting and taking each state and action. So far, we have assumed capacity for storing entries $v(s)$ and $q(s, a)$ for each state and state-action pairs in a look-up table manner. However, this is not feasible for environments with large number of states or actions (i.e. Atari games, states of which are characterised by image pixels, contain more different states than atoms in the universe [17]), and neither when a finite amount of experience is available. In these cases, the solution is to learn approximations of the state and action value functions using parametric functions:

$$\hat{v}(s, \mathbf{w}) \cong v_\pi(s)$$

$$\hat{q}(s, a, \mathbf{w}) \cong q_\pi(s, a)$$

where $\mathbf{w} \in \mathbb{R}^d$ is the set of weights of dimension independent to the number of states (and ideally much smaller). These methods are in the field of Value Function Approximation (VFA). As an alternative to VFA, Policy Gradient (PG) methods use parametric functions to directly estimate the policies:

$$\hat{\pi}(a|s, \theta) = Pr(A_t = a|S_t = s, \theta)$$

The goal when introducing w and θ in these frameworks is offering generalization. We look for parametric value functions that provide accurate estimation for unseen states and unexplored actions.

In comparison with VFA methods, PG methods show better convergence properties (although they converge to local optimums and not global) and are more effective in high-dimensional action spaces. However, PG method show higher variance when evaluating the learnt policies.

With this set up, Deep Neural Networks (DNNs) make its way to RL since DNNs are outstanding universal function approximators. When DNNs appear in RL we relate to deep RL (DRL). DRL has shown great results in tasks that could not be tackled with traditional RL (i.e. robotics or NLP) and have shown superhuman performance in videogame environments [18, 19, 20, 21].

2.3 Challenges

In the field of RL and DRL, there exist huge differences on the experimentation of different methodologies depending on the environment that is used. While some benchmark models can successfully tackle simple

environments, they can fail drastically in more complex ones. Arguably, the characterization of the complexity of RL environments depends on their underlying dynamics (which conditions the difficulty to explore them) and the shape of their associated reward distributions. For instance, a maze environment is more difficult to explore than an open field due to the environment dynamics. Hence, in the beginning of the training process in the maze environment, agents (with random behaviours) will struggle in reaching novel states that can give them valuable information. Hence, the problem with environment dynamics is that they can define very unbalanced distributions over states $s \in p(s)$. In the maze scenario, and assuming a dense reward distribution, agents would most probably fall into local optimums on the maximization of the reward. In addition to environment dynamics, the reward distribution associated to a given task in an environment conditions the learning process of the agents. Since the obtained reward is the only signal driving the learning processes of the agents, if the reward distribution is very sparse, meaning that rewards are provided in very few states, the agents will never be guided. In this sense, even if the agents benefit from a suitable distribution over states $p(s)$, they will struggle in assessing the value of the actions to take in each state because for that they need to obtain reward outcomes. In the real world context, many RL problems consist of hard-exploration problems (due to the complex dynamics of the real world) and shaped with sparse or reward-free distributions. For instance, designing an agent to navigate effectively from one room of an apartment to another one is difficult to explore due to the dynamics of the environment (the agent can only leave a room navigating through a little door) and the reward distribution is sparse since the agent only receives a positive reward when the navigation goal is reached. If we had an oracle that at each step told the agent how far is from the navigation goal, the reward distribution would be dense and the problem could easily be solved with DNNs. Reward shaping [22] for easing such tasks is possible in the real world if we wanted the agent to learn to navigate to a single goal. However, it would not solve guided navigation, where a user determines a different navigation goal at each episode (we will not put oracles in every point of our apartments), which would actually define a multitask problem. Even in the case that we put an oracle to every point in our apartment and we designed an agent to learn to navigate to all the positions, we would not be able to do so by defining dense distributions of rewards. That is because of catastrophic interference [23], a tendency of DNNs to abruptly forget previously learned information when given new one. This means that the agent would be capable of learning to navigate to one goal at a time, and would only support being shaped by a single reward distribution. Intuitively, this comes from the fact that the agents learn to assess the values of visiting states and taking actions depending on the reward outcome that they receive. In this sense, the value functions of their behaviours does not generalize nor scale (i.e. navigating to the sofa might be a valuable behavior for looking for a lost TV remote but will be rapidly forgotten when learning to close the bathroom door). Moreover, in the majority of cases, crafting a unique dense reward distribution that generalizes to multiple tasks within an environment demands too much expertise and is not feasible.

To overcome such limitations, intrinsic curiosity [24] and empowerment [25] methods provide intrinsic rewards that encourage the agents to seek for novelty and hence explore the unexplored, which increases the probability of eventually succeeding in the extrinsic tasks. These methods aim to overcome the limitations of RL in environments with sparse rewards by defining proxy rewards that can shape the agents in the early stages of their learning processes. These intrinsic rewards aim to enable the agents to learn baseline task-agnostic representations and behaviours that generalize and are valuable to perform any task within the environment.

With all this, we expect truly autonomous embodied AI agents to function in environments with sparse rewards or even reward-free. We dream with agents that approach artificial general intelligence so that they can learn multiple objectives in the same way as we humans do. For that, this thesis focuses on the

representation learning and self-acquirement of skills/abilities in the absence of task-oriented rewards.

3. Related Work

Reinforcement Learning (RL) [26] has witnessed a wave of outstanding works in the last decade. The ones that received more interest being in videogames [20, 19, 18], but also in robotics [27, 28]. These works follow the classical approach of RL. This approach consists of an agent that interacts with an environment and learns a policy that maximizes the expected sum of total rewards, namely the return. In the majority of cases, the reward distribution offers guidance to the agents in order to empower them to solve a particular task. This fact enables the agents to learn representations and policies end-to-end for solving the task that is tackled. However, the task-specific knowledge of these agents is not useful for being transferred to other downstream tasks due to catastrophic forgetting [23]. That is, the representations of the environment that the agents learn during the guidance of a specific reward signal only encode the value of each state for performing the given task (defined by the reward signal). This fact limits the scalability of the RL agents since they cannot approach multi-task problems or cannot obtain task-agnostic knowledge from an environment. For this reason, during the last few years the research community of RL has put lots of interest in the unsupervised and self-supervised approaches for learning task-agnostic representations [29, 30, 31, 32] and behaviours [33, 34, 35, 36, 1] within an environment.

In fact, the will to overcome supervision in AI and DL is not new at all. The Computer Vision (CV) and Natural Language Processing (NLP) fields have already experienced the self-supervised learning (SSL) revolution [37, 38, 39]. In these works, massive language models like the GPT-3 [39] and novel visual architectures like the Visual Transformer [40] achieve impressive results in multiple downstream tasks thanks to being trained in a generic task and in a self-supervised manner. Following the popularity of SSL, several paradigms are arising in the last few years for obtaining task-agnostic knowledge in RL [1, 36, 35, 41, 34, 33]. These works aim to uncover ways to motivate the agents to acquire valuable knowledge from an environment without supervision. In RL, the value of the task-agnostic knowledge can be assessed by both the generalization capabilities of the learnt representations and the usefulness of the learnt behaviours (policies). For this reason, recent works rely on the disentanglement of representation learning and RL [29, 1].

On the one hand, different approaches have arisen for self-learning valuable generic representations [42, 43, 31]. Some of these methods asses the quality of the learnt representations directly on the value that they provide to RL agents [29, 31]. However, RL environments that define high-dimensional state spaces (i.e. the pixel space) carry a set of difficulties that makes autonomous representation learning more difficult to accomplish [44, 41]. On the other hand, several paradigms have gained popularity for driving the RL agents learning of behaviours independently of extrinsic task-oriented reward functions. Curiosity-driven learning [24] is the approach that encourages the agents to seek for novelty in the environment and hence maximize the coverage of the state space [34, 45]. In this way, curiosity does not only provide capabilities for covering the state space in hard-exploration or task-agnostic environments [46] but also helps to drive learning in environments with very sparse rewards [34].

Empowerment has become an interesting and popular intrinsic curiosity framework [47, 25, 48, 49]. Empowerment relies on the information-theoretic approximation of the influence of a behaviour with respect to consequences in an environment. In this way, the empowerment of an RL agent consists of the maximization of the consciousness about the action-outcome interaction within a RL set-up. In several works that regard empowerment, the skills to discover and learn are defined as the behaviours that effectively perform such agent-environment interactions [1, 35, 50, 36]. In classical empowerment these interactions are between sequences of actions (behaviours) and final states [25]. Central to the formulation of empowerment is the

concept of mutual information (MI). Several works have already used this information-theoretic metric for driving the learning processes of RL agents in absence of task rewards [51, 52]. In comparison to classical empowerment [25], where an agent blindly commits to a sequence of actions, *Gregor et al.* [36] introduce a novel approach where the decision to take each of the actions also depends on the environment observation at that state. The latter is accomplished by maximizing the MI between states and some latent variables.

Our work is inspired from the Explore, Discover and Learn (EDL) [1] paradigm for the unsupervised skill discovery and learning of task-agnostic skills. EDL approaches the maximization of the MI between states and their latent representations [36]. However, EDL makes some assumptions that regard the MI formulation that do not scale to the pixel space. For instance, the fact that EDL uses variational inference [53, 42] involves the computation of the Mean Square Error (MSE) between states for computing the MI. Intuitively, the latter does not scale to the pixel space as one cannot relate the MSE between images to the actual distance in the environment.

To develop our work we deploy our solutions in the Habitat AI environment [2]. The Habitat simulator supports several datasets of 3D scenes. This work only makes use of the Matterport3D scenes available for Habitat [54], and tackles embodied visual navigation [55, 56, 57]. The current state-of-the-art methods for 3D navigation are based on Simultaneous Localization and Mapping (SLAM) [58]. These methods iteratively construct maps of the scene by memorizing the environment dynamics and observation resemblance through the intelligent usage of sensory data (does not always imply image data). The Habitat simulator provides an open-source SLAM-based baseline implementation together with an implementation of Proximal Policy Optimization (PPO) [59] algorithm. PPO is a RL-based policy gradient algorithm that combines learning from interaction with the environment and from the optimization of a surrogate objective function on collected batches of data. PPO has demonstrated to outperform other online policy gradient methods [59] in several RL environments. Since this work relies on RL, we will use PPO for implementing our experimentation.

This work focuses on image goal-driven navigation [60] (ImageNav). For tackling general RL goal-oriented problems, the most popular approach is to use Universal Value Function Approximators (UVFA) [61] where at each step, the agents take actions according to a policy which is jointly conditioned by the current agent state and the selected goal.

4. Goals and Problem Formalization

Acquiring abilities in the absence of a task-oriented reward function is at the frontier of RL research [1, 35, 25, 36].

The goal of this thesis is to extend the paradigm of Explore, Discover and Learn (EDL) to the pixel domain. The purpose of doing so is to provide an end-to-end framework for RL agents to discover and learn skills from image data in a self-supervised task-agnostic fashion.

The EDL paradigm intersects with empowerment: EDL addresses the reward-free skill discovery and learning tasks to discover *what* can be done in an environment and *how*. EDL consists of unsupervised skill discovery and training of RL agents without considering the existence of any extrinsic motivation or reward. The EDL ultimate goal is to learn state-covering skills in an unsupervised manner. It addresses the task by breaking down the end-to-end training process to three disjoint problems: (i) Exploration, which goal is to obtain a representative model of the state space; (ii) Skill-discovery, which goal is to relate the properties of the state space to a finite set of meaningful goals; (iii) Skill-learning, where RL agents are trained to maximize the mutual information (MI) between the current states and the state-covering skills. Each of these three stages can be studied, addressed and improved independently.

For the moment, EDL has been tested only in toy example mazes, so this work presents a pixel-based implementation of EDL available for embodied AI agents in Habitat. The objective of EDL is to discover representative state-covering goals and to learn RL policies, namely skills, that achieve the self-discovered goals. In this work, the discovered goals are interpreted as navigation objectives.

The Habitat set-up allows to configure many different tasks in the same apartment. These tasks are defined by different reward distributions depending on the goal that wants to be achieved. For this work, all the rewards will be discarded since the objective is to provide a framework for autonomous RL agents to obtain valuable knowledge in absence of task-oriented extrinsic rewards. In this way, for framing the underlying process of the environment in a way that allows RL agents to formalize their performance we consider a Markov Decision Process (MDP) without rewards. Hence, in this case the underlying MDP is defined as the tuple $\mathcal{M} = (\mathcal{S}, \mathcal{A}, \mathcal{P})$ where $\mathcal{S}$ is the high-dimensional set of states (defined by image pixels). $\mathcal{A}$ is the action space and $\mathcal{P} = p(s|s, a)$ is the transition function. The definition of the action space $\mathcal{A}$ is set to a finite set of actions that enables discrete movements. Concretely, the agents are only capable of moving forward 0.5 meters, turning left 45º, turning right 45º and not moving at all.

With this framework, the information-theoretic approach of EDL aims to learn latent-conditioned policies $\pi(a|s, z)$, and defines skills or options as the policies obtained when conditioning π on a fixed value of $z \in \mathcal{Z}$. We define $\mathcal{S} \sim p(s)$ as the probability function of visiting a state s (which is characterized by the image observation that the RL agents capture in that state), and $\mathcal{Z} \sim p(z)$ the probability function for the latent random variable $\mathcal{Z}$. In this way, for learning the skills EDL aims to find the policies that maximize the MI between $\mathcal{S}$ and $\mathcal{Z}$. Using notation from information theory, $\mathcal{I}(\cdot\,;\,\cdot)$ is the MI and $\mathcal{H}(\cdot)$ is the Shannon entropy. Due to symmetry, the MI can be defined in two different forms, namely the forward and reverse forms:

$$\begin{aligned} \mathcal{I}(S, Z) &= \mathcal{H}(Z) - \mathcal{H}(Z|S) \quad \rightarrow \quad \text{reverse} \\ &= \mathcal{H}(S) - \mathcal{H}(S|Z) \quad \rightarrow \quad \text{forward} \end{aligned} \tag{1}$$

As seen in equation (1), the maximization of the MI requires knowledge of the unknown distributions $p(s)$, $p(s|z)$ for the forward form and $p(z|s)$ for the reverse form. EDL uses the forward form of the MI for the maximization of the MI. The original EDL paradigm is applied to "toy mazes" where the state space is defined by 2D coordinates. In this way, Campos et.al [1] approximate $p(s|z)$ with variational inference and are then capable of computing the forward form of the MI with the differences between $p(s)$ and $p(s|z)$, which happens in the state space $s \in \mathcal{S}$. In this way, EDL uses the difference between the current state of the RL agents (defined by a 2D position) and the skill-conditioning goal state (defined by a discovered 2D position) to define a reward distribution which maximization results in the maximization of the MI itself. However, when working in the pixel domain the state space is defined by the high dimensional image pixels. Still, the objective is to achieve effective goal-conditioned navigation in an environment (which is 3-dimensional in this case). In this way, it makes no sense to relate the distance in the state space (difference between images) to the actual distance in the environment. For this reason, it makes more sense to aim to maximize the reverse form of the MI, which implies approximating $p(z|s)$ as a mapping of the state space $\mathcal{S}$ to a latent space $\mathcal{Z}$, choosing a suitable $\mathcal{Z} \sim p(z)$ and defining a reward that relates to the distances in the latent space $p(z) - p(z|s)$.

In the following, one can find the definition of the 3D environment used for experimentation. We majorly work over an specific scene of the Matterport3D [54] dataset available in the Habitat AI virtual environment, which will be related as the *white apartment*.

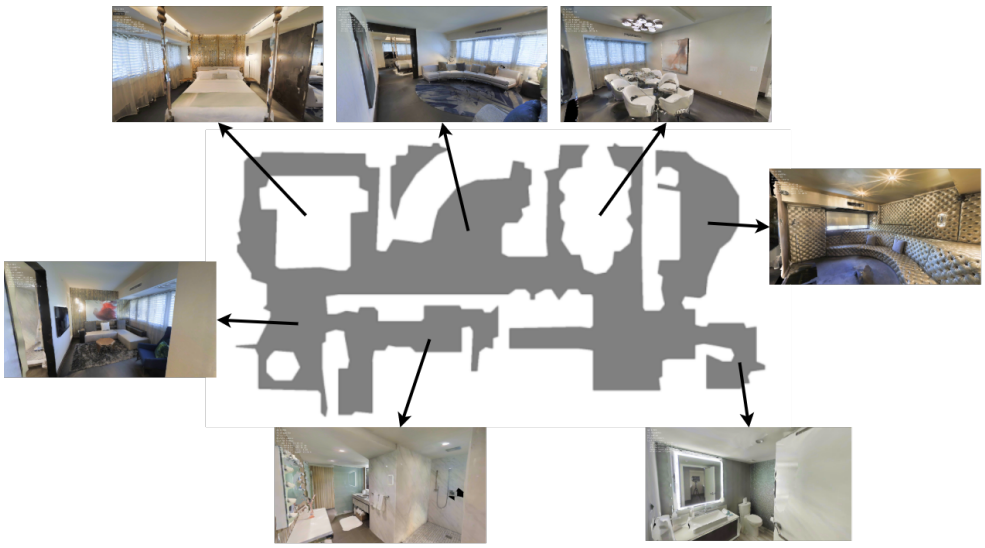


Figure 1: Top-down view of the white apartment.

As seen in Figure 1, the white apartment dynamics are complex. The apartment contains many different rooms and non-navigable areas within them. Furthermore, the apartment is rich in visual details and contains a considerable amount of variability among the visual appearance within the different rooms.

Figure 2: Samples from the interior of the white apartment.

Also, as seen in Figure 2 the apartment contains similar visual features in many different regions (i.e. there are a lot of white objects). Since the objective of our approach to EDL is to discover (and learn) valuable knowledge from image data only, the state space of the white apartment consists of image observations. Hence, our methodology will be tested in the available image data in the Habitat environment, which is shown in Figure 3.

Figure 3: The available image data in the Habitat environment: (left) RGB; (middle) instance segmentation; and (right) depth observations

5. Proposed Solution

In the following we characterize the pipeline that we follow for extending Explore, Discover and Learn (EDL) to the pixel domain. In general, we approach embodied visual navigation by leveraging self-supervised learning techniques, parametric policies and goal-conditioned reinforcement learning. We provide a methodology for *(i) exploring* a complete set of representative states; *(ii) discovering* generalist state-covering representations of the states and defining a set of meaningful navigation goals; and *(iii) learning* goal-conditioned policies that reach the navigation goals.

5.1 Exploration

Exploration is a central task in RL. The selection of the best actions in each state is guaranteed by previous exploration of all the available actions that can be taken in each of the available states within an environment.

Ideally, in a reward-free task agnostic set-up, one would like to learn baseline behaviours (skills) that are later useful for solving multiple unrelated downstream tasks. Without prior knowledge of the availability of useful skills within an environment, the objective that one can set is to at least ensure that these skills provide complete coverage of the entire state space [1, 45]. In this way, one would make sure that the learnt skills are not missing any information available in the environment. Hence, discovering state-covering skills requires complete exploration of the state space. Formally, the challenge of the exploration stage is to obtain a source of states $s \in \mathcal{S}$ that are representatives of the environment and that the learnt skills will ultimately cover. A reasonable choice for encouraging the discovery of state-covering skills is to perform sampling from $p(s) \sim \mathcal{S}$ uniformly. Uniform sampling would be possible if we assumed a considerable level of prior knowledge of the environment for which we could use an oracle to obtain different valid navigable states uniformly. A more realistic approach for tackling the inference of a state-covering distribution $p(s)$ is to train exploration policies to infer a uniform $p(s)$ [46]. However, as we deal with the high-dimensional pixel space, inferring a uniform $p(s)$ is not feasible. For this reason, the final approach is to adopt a non-parametric solution by collecting a dataset of trajectories in the environment. In this way, we tackle exploration by implementing random agents that perform episodes of a specific length and collect the trajectories of image observations.

As seen in Figure 4, the white apartment is too complex in its dynamics to be easily explored by random agents. For this reason, we will assume a little degree of prior knowledge of the environment for which we will set random navigable points for the agents to start in each of the episodes.

In Figure 5 it can be seen how with the aforementioned adjustment, random agents collect a dataset of image observations that covers a complete set of representatives of the environment.

5.2 Skill Discovery

The goal of the skill discovery phase is to extract a finite set of meaningful representatives of the state space. Concretely, following the formulation of the reverse form of the MI in equation (1), the objective of this phase is to model $p(z|s)$ as a mapping between the states s and some latents z, and $p(z)$ as a categorical distribution over the latents $z \in \mathcal{Z}$. Ideally, one would like the latent space $\mathcal{Z}$ to be low-dimensional and

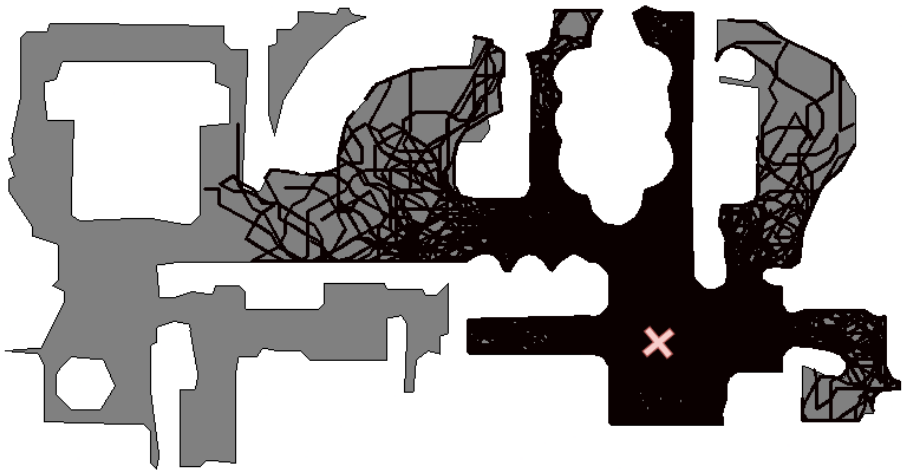


Figure 4: Results of the exploration performed by random agents in 500 trajectories of 150 steps in the white apartment. In all the episodes, the random agents start in the position marked by the red cross.

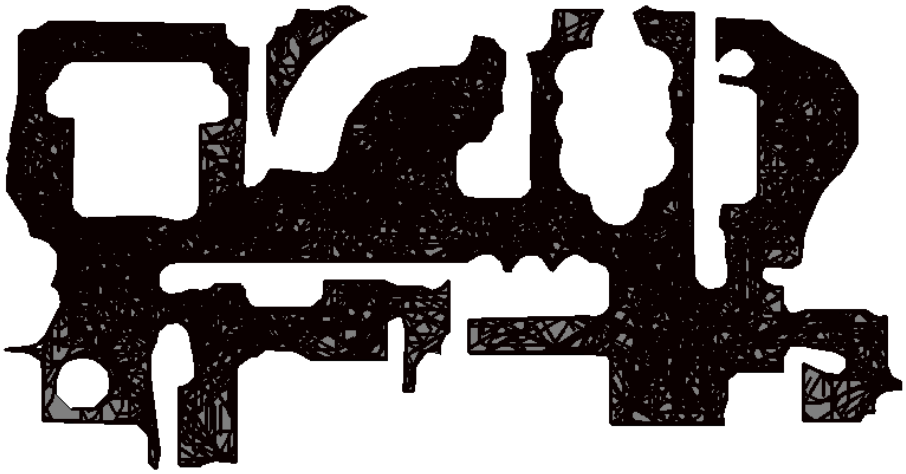


Figure 5: Results of the exploration performed by random agents in 500 trajectories of 150 steps in the white apartment. The random agents start in a random navigable position in each of the episodes.

to preserve valuable features of the state space. For this work, we aim to learn baseline navigation skills in the 3D Habitat environment. Hence, we would like to map the image observations s to latents z that encode both existing similarities in the images and the spatial relation between them.

This work studies two different approaches for mapping the image observations to meaningful representations. Since the two approaches are differnt, these imply further differences in the design of the marginal distribution of the latents $p(z)$. In the following, the two self-supervised learning approaches are characterized together with the pipeline for modelling both $p(z|s)$ and $p(z)$.

5.2.1 Contrastive

It is a SSL approach in which a siamese model (which uses two architectures in parallel) learns to define an embedding space by projecting pairs of positive and negative input examples (in our case of inputs images). Some examples of positive pairs of input images consist of augmentations of an original image (i.e. random crops, color changes, rotations) as it is done in CURL [31]. Then, negative pairs of input images are just pairs of different images of the dataset. In this way, contrastive-based approaches aim to characterize an embedding space that preserves the features of the images by projecting similar observations closer in the embedding space.

The training process of a contrastive approach consists in feeding a batch of original images to the main encoder and its corresponding positive and negative pairs to the second encoder (namely, momentum encoder). In a batch of N original images, the momentum encoder is fed with $N-1$ negative images and a positive one. In this way, the contrastive training process becomes a classification problem where the network weights are updated according to correct and incorrect predictions on positive or negative pairs. Both encoders consist of the concatenation of some convolutional layers that extract visual features with a small fully connected head (a small MLP). This work implements the contrastive approach by making use of the adaptation to CURL proposed by Stooke et al. [29], namely Augmented Temporal Contrast (ATC). Compared to CURL, in ATC the positive pairs of inputs consist of two image observations belonging to the same exploration trajectory. In this way, we encourage the model to map the image observations to an embedding space which encodes spatial relations between them. That is, we train a contrastive model so that a positive pair of inputs consists of two observations of the same trajectory with a delay $d \sim \mathcal{N}(\mu, \sigma^2)$. We experiment with $\mu = 15$ and $\sigma = 5$. Hence, in each epoch, each of the original images in the dataset is matched with a different positive pair (since d is sampled from a non-deterministic distribution). In this way, we perform a data augmentation in the temporal domain. With this configuration we obtain the latents $z, z' \in \mathcal{Z}$ in the outputs of the MLP heads. This siamese contrastive approach minimizes the $InfoNCE(z, z')$ loss for training the two encoders, which results in the maximization of the mutual information between the observations s and the latents z to which they are mapped [43**?**]. Formally, the latents z and z' are obtained after passing the pair (s, s') through the main and momentum encoders e_θ and e'_θ respectively and through their corresponding MLP heads h and h'.

$$z = h(e_\theta(s))$$
$$z' = h'(e'_\theta(s'))$$

Then, the network is updated according to the $InfoNCE(z, z')$ values.

$$InfoNCE(z, z') = log \frac{exp(z^T W z')}{exp(z^T W z') + \sum_{i=0}^{K-1} exp(z^T W z^i)} \tag{2}$$

Where W is a learned parameter matrix that allows the similarity computation between latent variables: $sim(z, z') = z^T W z'$.

At this point, given a model of $p(z|s)$ provided by the main encoder, we require a model of $p(z)$ for later computing the reverse form of the MI 1. Taking into account that the sampled latents $z \sim p(z)$ are the ones that will condition the skills $\pi(a|s, z)$, we want to model a categorical distribution $p(z)$ defined by a

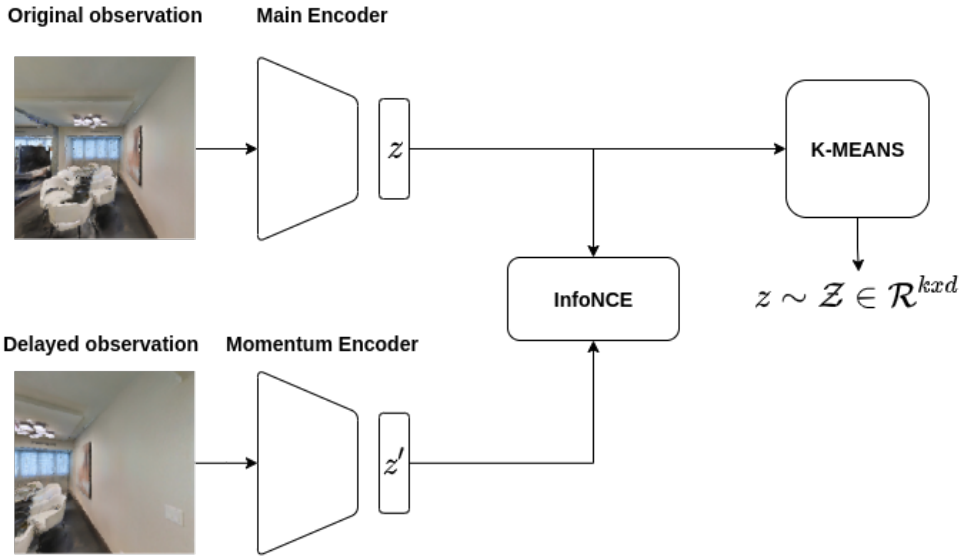


Figure 6: Schema of self-supervised contrastive approach for learning valuable features from image observations. A clustering in the embedding space is applied when the training loss is minimized to obtain k representative latent variables.

finite set of latents z. Intuitively, we want the selected latent variables to be representatives of the latent space $\mathcal{Z}$. With this, we assume that a good model of $p(z|s)$ will cause the z that are more representative of the latent space to be also representatives of the original state space in terms of existing environment similarities and spatial relations. A reasonable choice for obtaining these representatives is to perform a clustering in the embedding space [30]. For this reason, this work explores the application of K-means to finally obtain the finite set of latent variables z that define the categorical distribution $p(z)$. Figure 6 summarizes the contrastive approach for learning meaningful image representations and discovering goals in a self-supervised way.

5.2.2 Reconstruction

In this work, our self-supervised reconstruction approach performs variational inference by means of an encoder-decoder architecture. Variational inference is a well-known method for modelling posterior distributions without relying on the computation of marginal ones (i.e. $p(s)$), which is an advantage.

Concretely, we make use of a variational autoencoder (VAE) with categorical classes, namely vector quantised VAE (VQ-VAE). The VQ-VAE model uses an encoder to learn $q_\theta(z|s,\theta)$ and a decoder to learn $q_\theta(s|z,\theta)$. With this, the objective is to achieve good approximations of the actual distributions:

$$p(z|s) \approx q_\theta(z|s,\theta)$$
$$p(s|z) \approx q_\theta(s|z,\theta)$$

We use convolutional layers for obtaining the input image features in the encoder of the VQ-VAE and a small MLP to obtain the actual latent belonging to the input image. Then, the latents are mapped

to a discrete dictionary of embeddings, namely the codebook of the model. Finally, the decoder receives a specific embedding from the model's codebook and reconstructs an image that is consistent with all the images that have been mapped to the input embedding. The number, of embeddings in the model's codebook is an hyperparameter of the architecture. The schema of the VQ-VAE architecture is shown in Figure 7.

The approach for achieving good models of the mappings between image observations and latent variables (and viceversa) relies on the minimization of a reconstruction error together with an introduction of a commitment cost. Following the information-theoretic approach, introducing a commitment cost in the loss function encourages the model to maximize the perplexity. The perplexity expresses the uncertainty in the choice of an embedding in the model's codebook where to map the input images. In this way, we want to obtain (and actually use) embeddings in the model's codebook that encode disjoint regions of the state and latent space. In this way, the overall loss in the reconstruction approach is defined as follows:

$$\mathcal{L}(s, s') = log(p(s'|z_q(s))) + \beta\|z_e(s) - sg[e]\|_2^2 \tag{3}$$

The first term of the loss belongs to the reconstruction error, and measures how capable is the model of reconstructing s' from the quantized encoded input observation $z_e(s)$, with z_e being the encoder and z_q the decoder. The second term ensures that the values of the encoded observations do not grow and commit to an embedding from the model's codebook (sg means *stop gradient*). Finally, β weights the importance of the second term in the overall loss.

With the reconstruction approach, the definition of $p(z)$ as a categorical distribution over the latent space is straightforward. In our case, we obtain $p(z) \sim \mathcal{Z}$ by defining a uniform distribution over the embeddings in the model's codebook. This makes sense because the embeddings in the model's codebook have been defined to be disjoint and to preserve the characteristics of different sets of input images, which intersects with the definition of the cluster centroids of the contrastive approach.

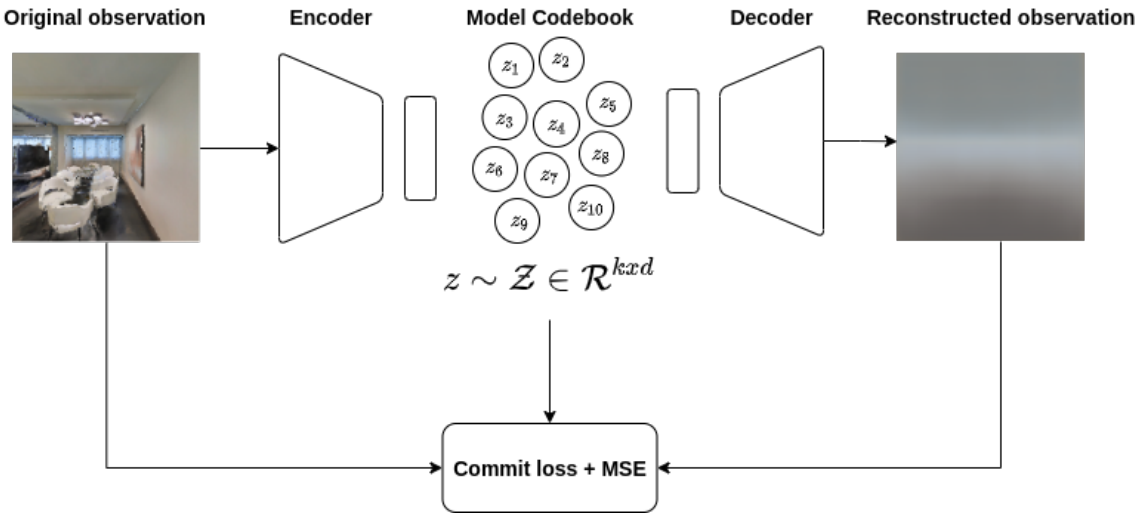


Figure 7: Schema of self-supervised reconstruction approach for learning valuable features from image observations. We apply a clustering in the embedding space when the training loss is minimized to obtain k representative latent variables.

The VQ-VAE model relies in the sensible value of the commitment cost for configuring the model's codebook [62]. For this reason, we implement an initialization technique that smoothes the dependency between the commitment cost and the convergence of the model codebook. Particularly, we forward all the collected image data to a randomly initialized encoder architecture and perform a K-means clustering of the embedding space. With all this, we initialize the model codewords to each of the found centroids. In this way, the model starts tunning the encoder/decoder and codebook from a disjoint set of representatives. We find that this technique allows to relax the hyperparameter dependency of the VQ-VAE model.

5.3 Skill Learning

In the last stage of the training process, the goal is to learn latent conditioned policies, namely skills $\pi(a|s,z)$, that maximize the mutual information between the states $s \in S$ and the latents theirselves $z \in \mathcal{Z}$. In this way, the embodied RL agents learn multiple self-discovered intrinsic objectives. In this work, the intrinsic objectives are interpreted as navigation goals. Hence, our final objective is to learn baseline navigation skills on self-discovered conditioning goals. A successful skill learning phase would provide the RL agents with the capabilities of effective navigation towards the regions of the state space encoded by the representative latents $z \in \mathcal{Z}$.

At this point, the training pipeline of the skills does not rely on the way in which the latent variables have been obtained (i.e. either with the contrastive or reconstruction approaches in our case). Given the latent variables z, a reasonable choice for training the skills is to define $z \sim p(z)$ as a uniform distribution from which we will sample at each training episode. In this work we assume that all the discovered skills are equally important and well-defined so a uniform $p(z)$ might be suitable for learning the multiple skills. Another more complex option would be to infer the $p(z)$ distribution so that we would sample the conditioning latent variables with probability depending on the importance of these (i.e. importance of a skill could be defined by the size of the specific cluster covered by the sampled latent).

The implementation of the training pipeline of the RL agents and policies makes use of the PPO [59] architecture provided by the Habitat environment. At each step, we feed the concatenation of the encoded image observation of the current step together with the conditioning z to the PPO network. Then, PPO outputs a distribution of probabilities over the actions and selects one by sampling from it. Now, to encourage the maximization of he MI between states and latents we craft a reward distribution which maximization coincides with the maximization of the MI itself. As mentioned in Section 4, we use the reverse form of the MI to assess the differences in the image-based state space. For this reason, we craft the rewards in equations (4) and (5) for the reconstruction and contrastive approaches respectively:

$$r(s, k=z) = \begin{cases} 1, & \text{if } k = \text{argmin}_j ||z_e(s) - e_j|| \\ 0, & \text{otherwise} \end{cases} \tag{4}$$

$$r(s, k=z) = \begin{cases} 1, & \text{if } k = \text{argmax}_j ||z_e(s) W e_j|| \\ 0, & \text{otherwise} \end{cases} \tag{5}$$

Where s is the current observations of the RL agents, $z \sim p(z)$ is the conditioning goal representation and k is the index of z within all $p(z)$. In this way, we only give rewards to the agents if they commit to actions that take them to a next state where the obtained image embedding $z_e(s)$ is closest to the conditioning z

embedding among all $z \sim p(z)$. Hence, with these definitions we are giving and maximizing a reward by computing a distance in the embedding space as $z - z_e(s)$ which coincides with the reverse formulation of the MI in equation (1). Note that for the contrastive approach, $z_e(s)$ is the result of passing the image observation s through the trained main encoder, and for the reconstruction approach it is the result of passing s through the VQ-VAE encoder. Figure 8 shows the skill learning pipeline for both the contrastive and reconstruction approaches.

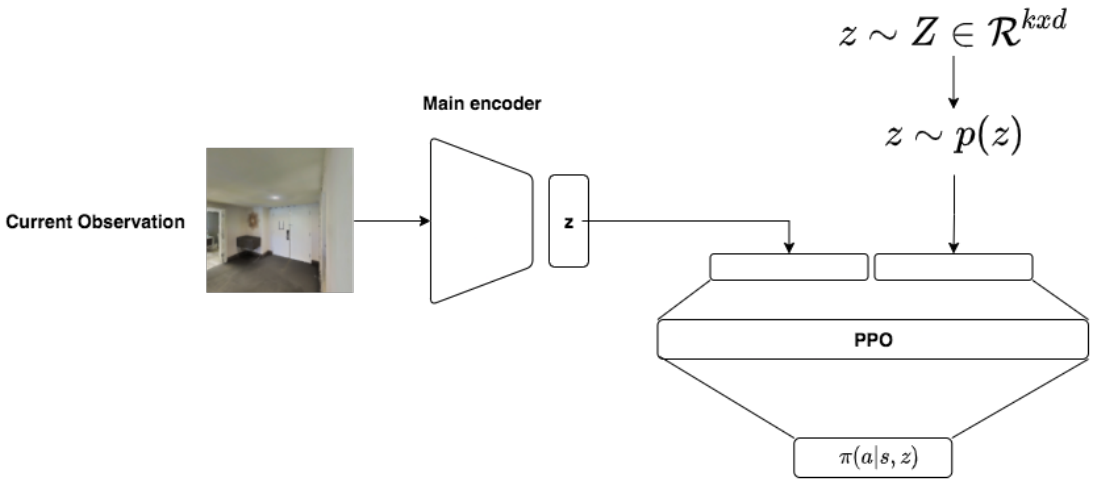


Figure 8: Schema of the skill learning pipeline. The pretrained main encoder is used for obtaining the input image latent representations. Then, the input latent is concatenated with the sampled discovered goal latent and fed to the PPO network all together. PPO outputs an action distribution for the current input state.

We refer to the task of learning the skills as *SkillNav*, which is a navigation episodic task. Formally, we define the *SkillNav* task as an episodic navigation task where at each episode we sample a navigation goal $z \sim p(z)$ uniformly and we train the agents to reach the goals within 150 steps (which is consistent with the number of steps of the random agents performing exploration in the apartment). Then, we assess the performance of the agents by evaluating the reward curves that they obtain in addition to the evaluation of policies that they learn (which we can see in generated videos). Moreover, we obtain insights of the performance of the training process by looking at the *Success* and *Success weighted by Path Length (SPL)* metrics provided by Habitat. The success measure is only positive when the agents call the *stop* action (which does not move the agents at all) in a distance less than 1 meter away from the sampled navigation goal in the current episode. Then, the SPL measures the ratio of the obtained success signals with the length of the navigation paths performed, hence assessing not only the capabilities to reach the goals but also to do it optimally. Finally, for PPO we can evaluate the entropy of the distribution of probabilities of the actions belonging to the policy. Intuitively, we expect the entropy to reach lower values during training since the agents reduce the uncertainty in the actions that they take, hence defining a robust behaviour. In this way, the entropy of the distribution over actions should reach the maximum in the early stages of the training process when the agents adopt random behaviours and the uncertainty in their decisions is very high.

6. Results

In this section we present and discuss the obtained results of the skill discovery and learning of our methodology. With this, we aim to provide insights on the process of self-supervised learning from pixels. We test the quality and assess the value of the learnt representations for RL agents that navigate in 3D environments. Before that, we first show in Figure 9 the results of the skill discovery stage of the standard EDL [1] in the white apartment of the Habitat environment. In the case of EDL, since the skills are learnt by a VQ-VAE model that works over spatial coordinates within the state space, the skills only encode spatial relations over the environment. Hence, the found clusters do not provide a segmentation of the state space that encodes the existing visual similarities in the state space. The latter can be seen in Figure 9 where two different rooms that look very different are mapped to the same cluster, or where the same room is split in two, even though it looks the same visually. For leveraging the embedded visual information in realistic 3D environments we implement the self-supervised representation learning pipelines described in Section 5.2 and show their results in the following.

Figure 9: Representations learnt by a VQ-VAE model on environment position coordinates.

6.1 Skill Discovery

In the following section we assess the quality of the learnt representations during the skill discovery phase for both contrastive and reconstruction approaches. Each experiment is characterized by the input data that is used (i.e. RGB, depth, instance segmentation).

6.1.1 RGB sensor

The RGB sensor captures colored images of the environment and stores them in objects of size *(image width, image height, channel)* where the channel is 0 for R, 1 for G and 2 for B. Each of these channels contain integer numbers to define the color values. For the following experimentation, we use 500 episodes of 150 RGB observations.

Using RGB data only we obtain the segmentation of the state space shown in Figure 10:

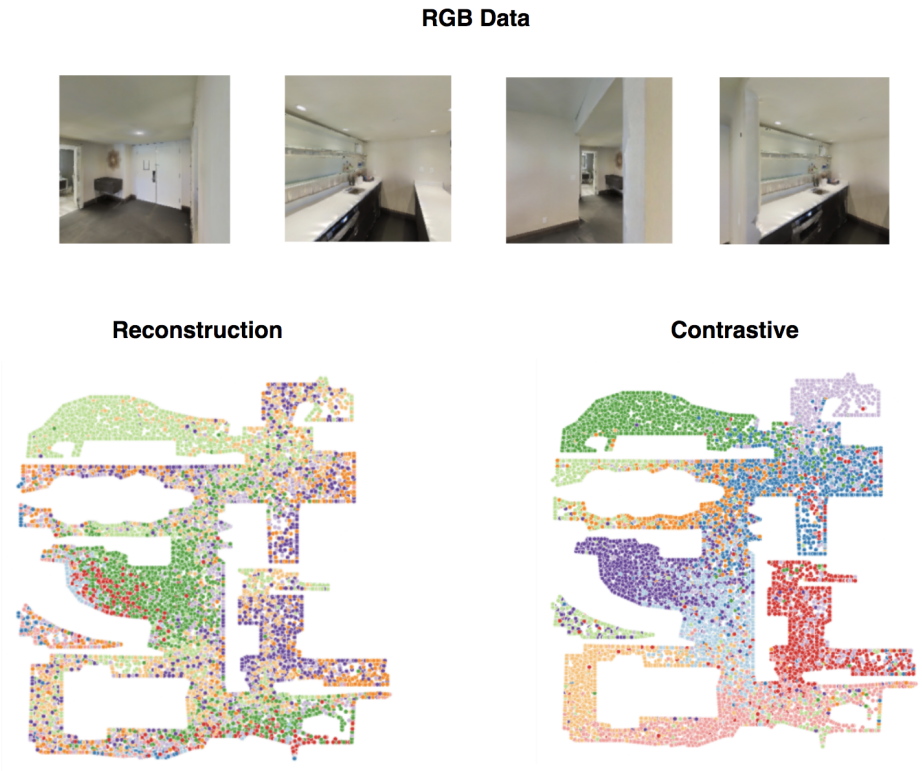


Figure 10: Index maps obtained with RGB input data only.

We refer to the colored plots in Figure 10 as *index maps*. Index maps are obtained after computing the centroids of the clusters of the embedding space and projecting the encoded images in the dataset of collected trajectories to the closest centroid.

As it can be seen in Figure 10, the contrastive approach is capable of obtaining a meaningful spatial segmentation of the environment only from RGB data. The contrastive-based model finds 10 almost equally represented clusters that encode quasi-disjoint regions of the stats space. In this way, we obtain a model of the environment that encodes both existing similarities in the image observations and spatial relations among them. Also, the clusters found by the reconstruction approach coincide in some cases with the ones found by the contrastive one, but look much more overlapped and less disjoint. The fact that the discovered zones are not clearly separated is a problem for interpreting these as navigation goals, since the agents would receive positive rewards in different zones of the apartment and would not converge to a spatial location.

Figure 11: Reward map obtained with the contrastive approach and RGB data in the white apartment.

Following the reward definition in equation (5), we obtain the reward distribution shown in Figure 11. We refer to this type of figures as *reward maps*. In the reward maps we see the zones where the agents will receive a positive reward for each of the discovered goals. For instance, if we conditioned the RL agents with the goal representation of the green cluster of the contrastive approach in Figure 10, these agents would only get positive rewards if they navigated on the yellow zone shown in the second map in Figure 11.

In Figure 12 it can be seen how the reward distribution formulated in equation (4), which is obtained by the reconstruction approach does not provide clear and separate navigation goals in general. However, see that the second goal representation in the contrastive approach coincides majorly with the third one obtained by the reconstruction approach.

Figure 12: Reward map obtained with VQ-VAE and RGB data in the white apartment.

Furthermore, for the reconstruction approach we can show the images that the VQ-VAE model reconstructs

for each of the latents $z \sim \mathcal{Z}$. These are shown in Figure 13. We can perceive some variability in the reconstructed images that indicates that these encode visually different zones of the apartment. They do not provide much detail since these images are reconstructions of clusters by definition.

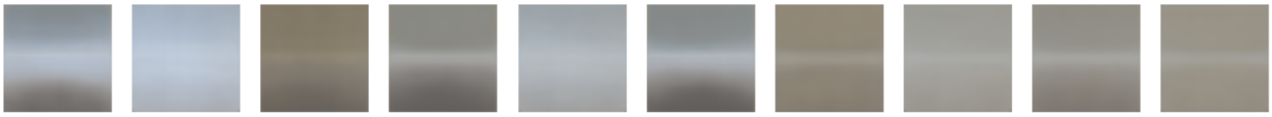


Figure 13: Reconstructed images from the VQ-VAE codebook on RGB data.

With all this, the contrastive approach shows much better capabilities than the reconstruction approach for learning meaningful representations when only RGB data is considered.

6.1.2 Depth sensor

The depth sensor stores one-channel objects of size *(image width, image height)* that accurately encode the depth of the perceived environment. These objects store the depth values with decimal variables (i.e. floats), and hence they involve a much higher cost in storage. For the following experimentation we use 500 episodes of 150 depth observations.

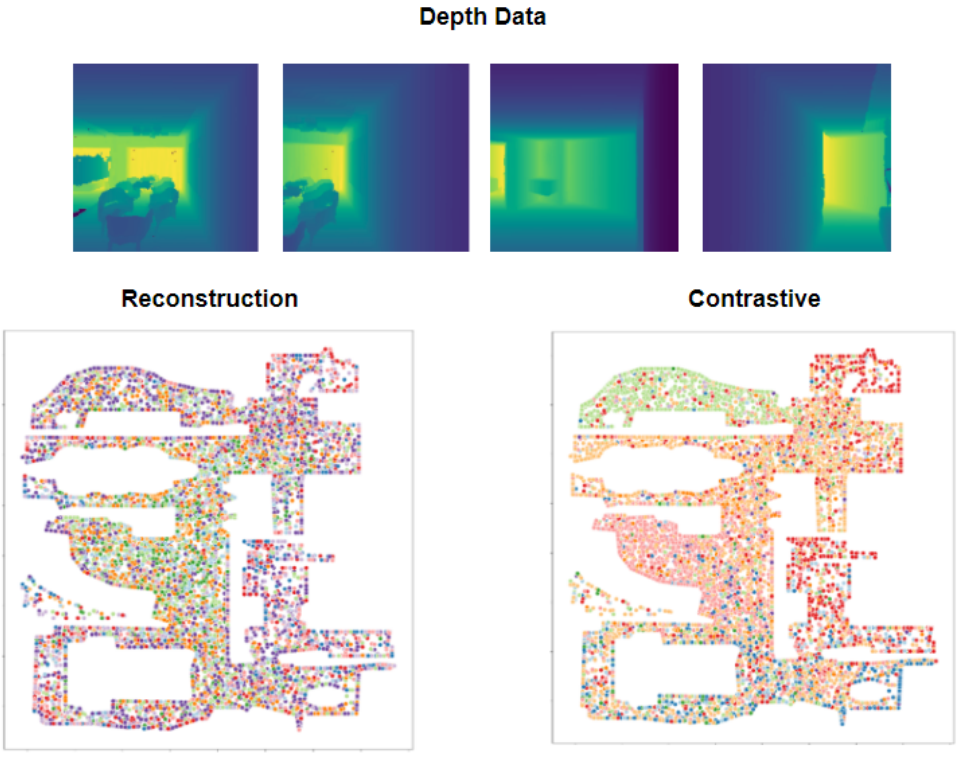


Figure 14: Index maps obtained with depth data only.

Using only the depth channel of the image observations we obtain the index maps shown in Figure 14. It can be seen that both the contrastive and reconstruction approaches struggle in finding meaningful and

disjoint areas that can encode navigable goals. Although, the contrastive approach shows better capabilities than the reconstruction approach for finding more clear clusters. Even though the visual information that is encoded in the depth sensor does not show to be enough for discovering clear navigation goals, we consider that the sensor provides valuable information that can be leveraged all together with the RGB sensor. Then again, the contrastive approach shows to be robust on learning representations from pixels in a self-supervised way.

Figure 15 shows that the perplexity metric is adequately maximized in the training of the VQ-VAE so that the model commits to all the latent representations in its codebook (defined in Section 5.2.2). This fact might indicate that the learnt representations not being meaningful is not being caused by a wrong training configuration.

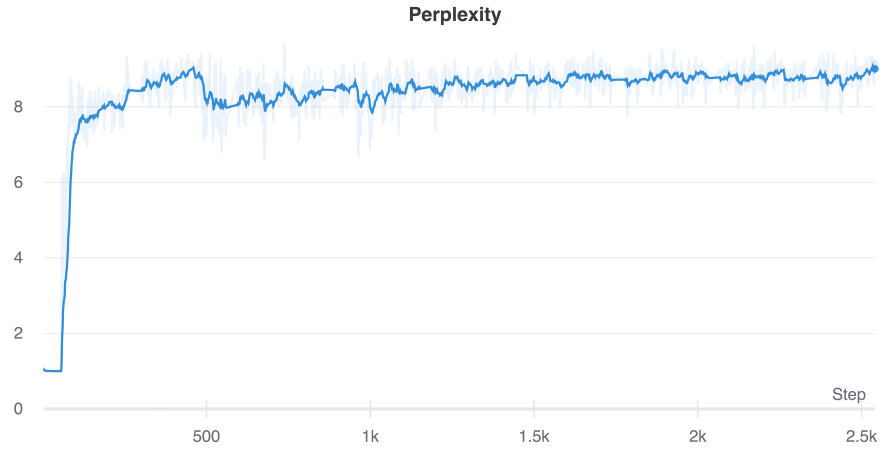


Figure 15: Perplexity curve of the training process of the VQ-VAE model when fed with depth data only. The model commits to the all the possible model codewords.

The reconstructions of the latents in the VQ-VAE model when fed with depth data only is shown in Figure 16. With these reconstructions it can be seen that the VQ-VAE model is not capable of learning disjoint representations of the image observations with respect to their depth channel variability.

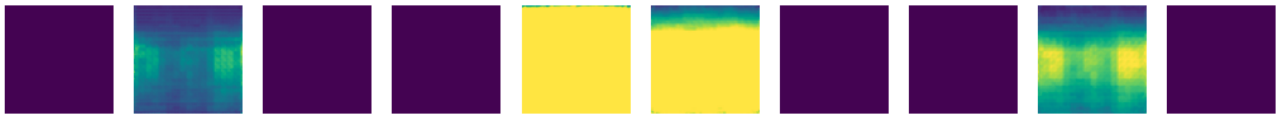


Figure 16: Reconstructed images from the VQ-VAE codebook on Depth data.

6.1.3 Instance segmentation sensor

In the following we make use of the image data that encodes the segmentation of instances within the environment. The instance segmentation consist of one-channel objects of size *(image width, image height)* where the object masks are encoded with integer values.

The results of the skill discovery stage are shown in Figure 18.

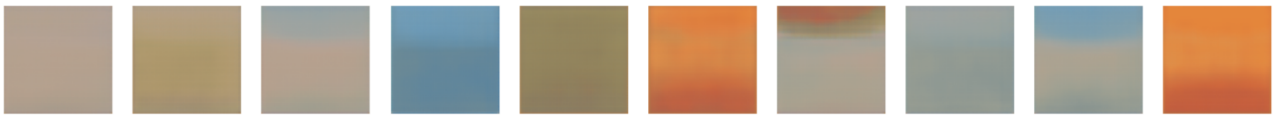


Figure 17: Reconstructed images from the VQ-VAE codebook on instance segmentation annotations.

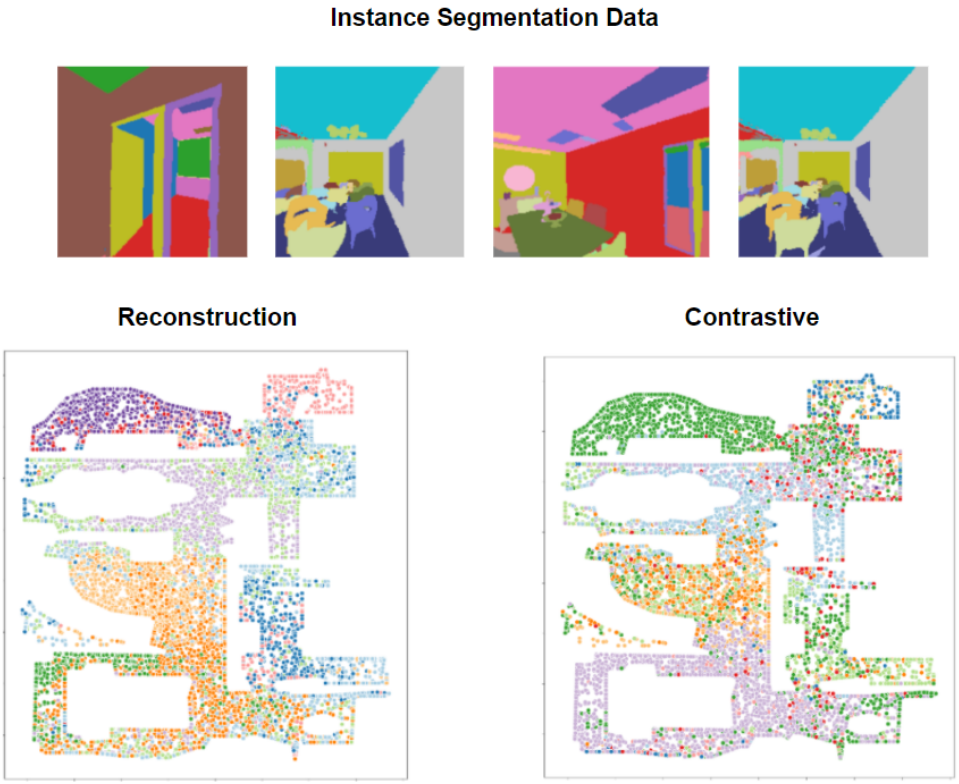


Figure 18: Index maps obtained with instance segmentation annotations only.

With the segmentation of instances we leverage the masks that encode different objects within the environment. In this case, we can see that this type of image data provides features that are more valuable than the depth channel information for discovering disjoint spatial zones. It can be seen in Figure 18 that both reconstruction and contrastive approaches discover some interesting and meaningful areas, and in some cases these areas coincide in both approaches. When dealing with instance annotations, we do not consider that one approach is more valuable than the other.

Since we see that some valuable information is encoded in these annotations we motivate further experimentation on skill discovery using the joint information provided by the RGB images, the depth channel and the instance annotations. Furthermore, we hypothesise that not only the segmentation of instances is useful for discovering disjoint representations of the environment, but object (class) segmentation would provide a better understanding of the scene. Figure 19 shows the difference between instance segmentation and ground truth masks. The key difference is that in instance segmentation, a pair of objects of the same class (i.e. chairs) are encoded differently. It can be seen that object segmentation encodes all the walls with the same color which does not happen in instance segmentation annotations. Intuitively, we consider

that more meaningful differentiation of the state space would be provided by object segmentation.

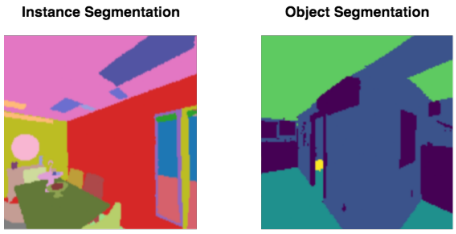


Figure 19: Difference between instance and class annotations.

6.1.4 RGB + Depth + Instance Segmentation

Finally, we leverage all the visual information provided by the Habitat environment in order to discover representative navigation goals within the environment. For this reason we train both the contrastive-based and VQ-VAE models to receive 5-dimensional inputs consisting of the stack of 3 RGB channels plus 1 for the depth channel and 1 for the instance segmentation annotations. With this settings, we do not modify the models or adapt them to receive different data types all together. Due to the limitation in the storage capacity of the server used for experimentation we are only capable of training the models on 200 episodes of 150 5-dimensional observations.

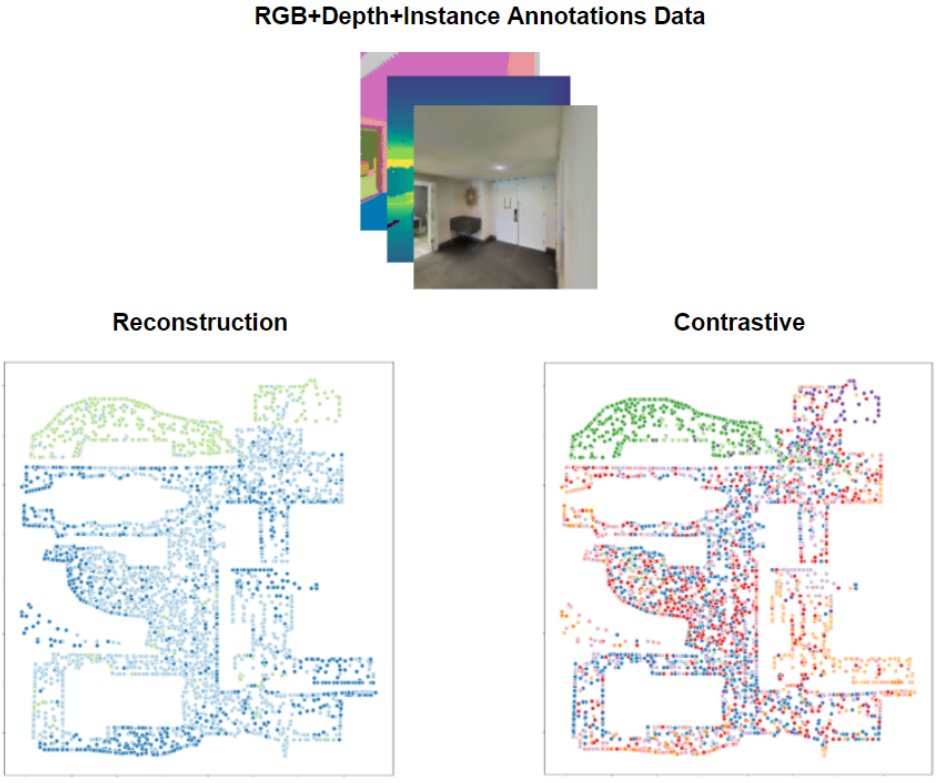


Figure 20: Index maps obtained with RGB, depth and instance annotations data.

In Figure 20 it can be seen that providing too much information to the models does not improve the skill discovery task. Concretely, the VQ-VAE model struggles with committing to more than 3 codewords, even though some clusters are meaningful. Also, the contrastive approach does not discover navigation goals as well-defined as with RGB data only. For this reason, we consider that due to the lower performance and the cost of obtaining depth and instance annotations, making use of RGB observations is enough for discovering meaningful navigation goals. However, we hypothesise that more sophisticated reconstruction and contrastive architectures could leverage all the visual information available in a more accurate way. For example, instead of sharing the model parameters for encoding the visual features of the images, making use of separate encoders for each type of image data could lead to better results.

6.1.5 Generalization among different apartments

In the following we evaluate the generalization, robustness and consistency of the contrastive and reconstruction approaches. For that, we apply the contrastive-based and VQ-VAE models to other apartments in the exact same way that we do in the white apartment and evaluate if the models are robust. Ideally, we are interested in that they do not demand high level of fine-tunning, and that they discover meaningful goals independently of the complexity of the apartments. In Figure 22 we show the results of both contrastive and reconstruction approaches for what we refer to as the *maze* apartment due to its complex dynamics. Without any changes in the architecture of the two models, it can be seen that the contrastive approach is robust and allows to discover meaningful goals also in the complex maze apartment. Also, the reconstruction approach finds some interesting patterns but the areas that we want to interpret as navigation goals are much more overlapped and less clear.

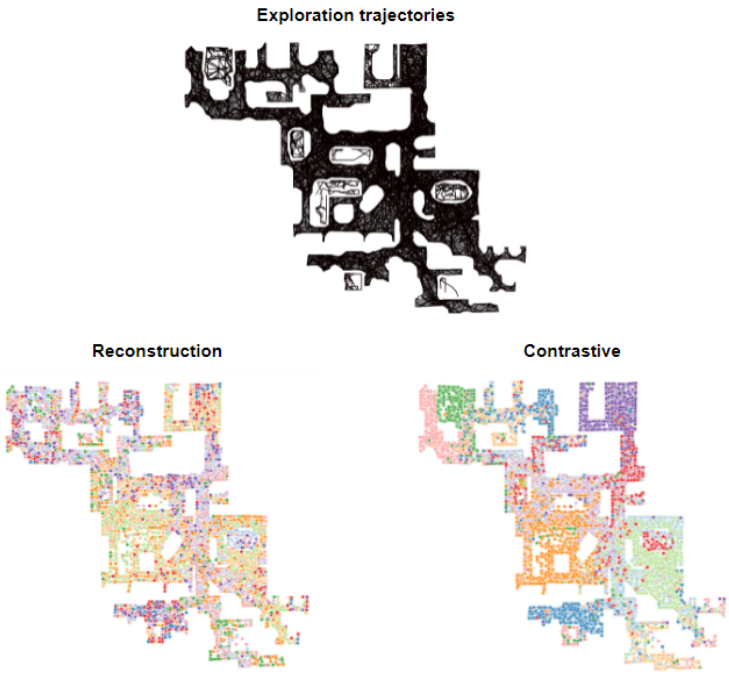


Figure 21: Exploration trajectories and index maps on the maze apartment.

We also test the results of the skill discovery phase in what we refer to as the *circular* apartment and show

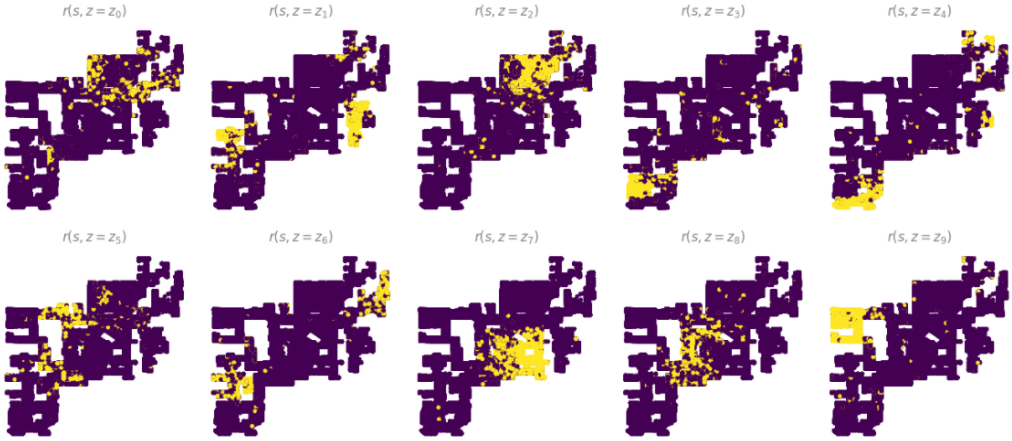


Figure 22: Reward distribution (5) provided by the representations learnt by the contrastive approach in the maze apartment. Many navigation goals (i.e. z_2, z_3, z_6, z_7 and z_8) are well defined and not overlapped.

the results in Figure 23. Surprisingly, the reconstruction approach is not capable of committing to more than one codeword. Again, the contrastive approach proves to be robust and provides a clear segmentation of the state space.

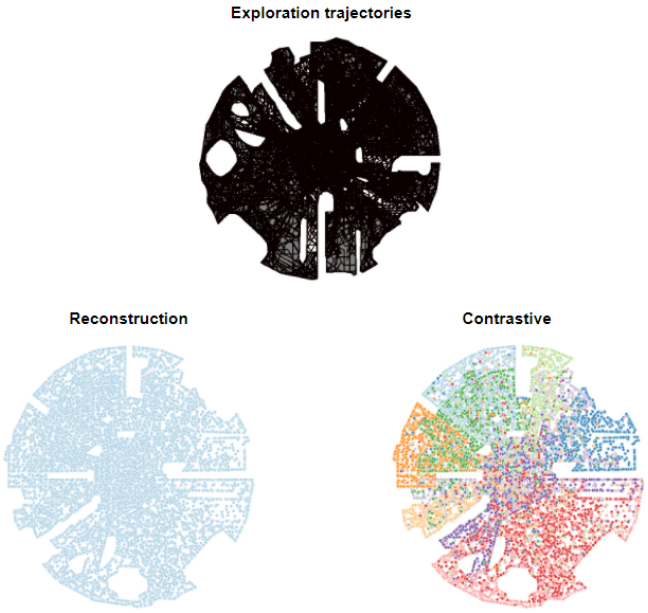


Figure 23: Exploration trajectories and index maps on the circular apartment.

Overall, we find that the contrastive approach is more robust than the reconstruction one because it provides high-quality representations independently of the visual appearance and complexity of the environment, and independently of the data type that is used for training the models (i.e. RGB, depth, instance annotations).

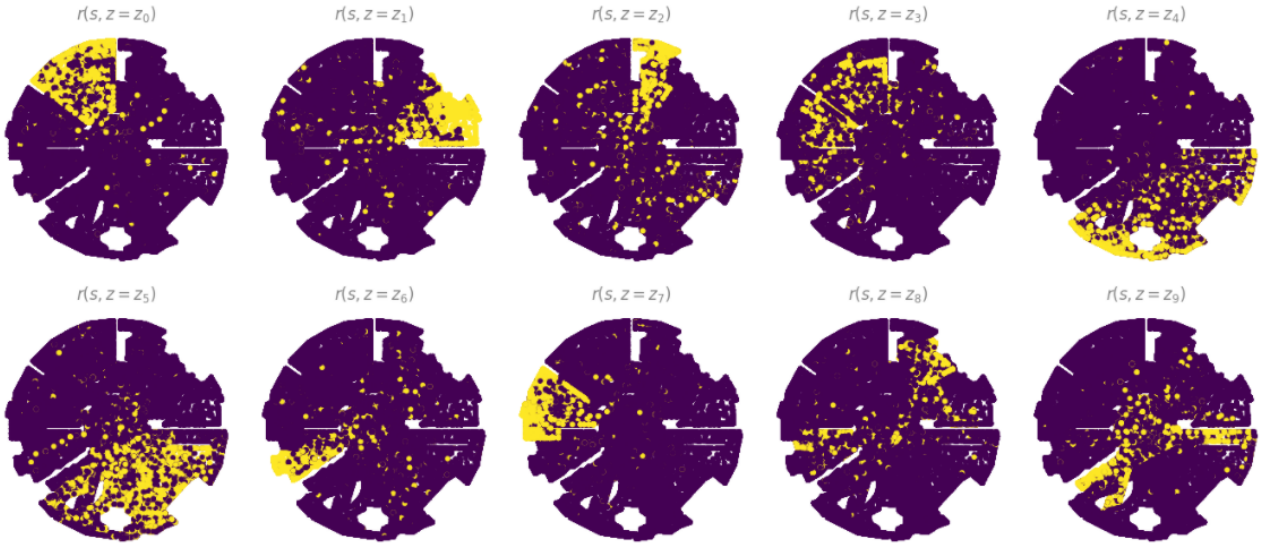


Figure 24: Reward distribution (5) provided by the representations learnt by the contrastive approach in the circular apartment. In general, the reward distribution encodes consistent navigation goals.

6.2 Skill Learning

In the stage of skill learning we interpret the discovered goals as navigation objectives. With this definition, we aim to learn latent-conditioned policies (skills) $\pi(a|s, z)$. For that, we make use of the contrastive-based model encoder that we have pre-trained with RGB data since it provides an accurate segmentation of the state space (see Section 10). Hence, when conditioning with a specific goal z we concatenate the encoded image observation of the current step with z and feed it to the PPO network (see Section 5.3). Finally, at each step we give a reward of 1 only if the embedding of the current image observation is closest to the goal z among all $z \in Z$ as defined in equation (5). Overall, we train the agents for 3,000,000 frames and obtain the curves in Figure 25.

In Figure 25 we see the reward curve increase to an average value of 115, which is promising taking into account that the maximum reward that can be obtained each episode is 150 considering 1 reward for each step. Also, we see that the average distance to goal remains stable in the value of 5,5. This indicates that the agents never really reach the navigation goals which are the centroids of the discovered clusters. Moreover, the success and SPL metrics decrease as the average reward increases which correlates with the fact that the agents are not succeeding in reaching the navigation goals but are obtaining positive rewards before that. Finally, we see the expected result in the entropy of the distribution of the agent's policy for which the agent commits to a more clear behaviour which decreases the uncertainty of the actions that it takes. In this way, Figure 25 shows the results of what could be a promising skill learning phase. However in Figure 26 we now show the reward obtained in evaluation time for each of the 10 goals.

The fact that the agents obtain reward when tackling skills 0 and 6 is not surprising since these skills provide positive rewards in the initial location where the agents start (see Figure 11). What is more concerning is the fact that the agents are not capable of learning the other more interesting skills for which they have to actually navigate to reach the goals. We hypothesise that even though the reward distribution in Figure 11 seems reasonable, it might provide too sparse signals to the agents for complicated navigation

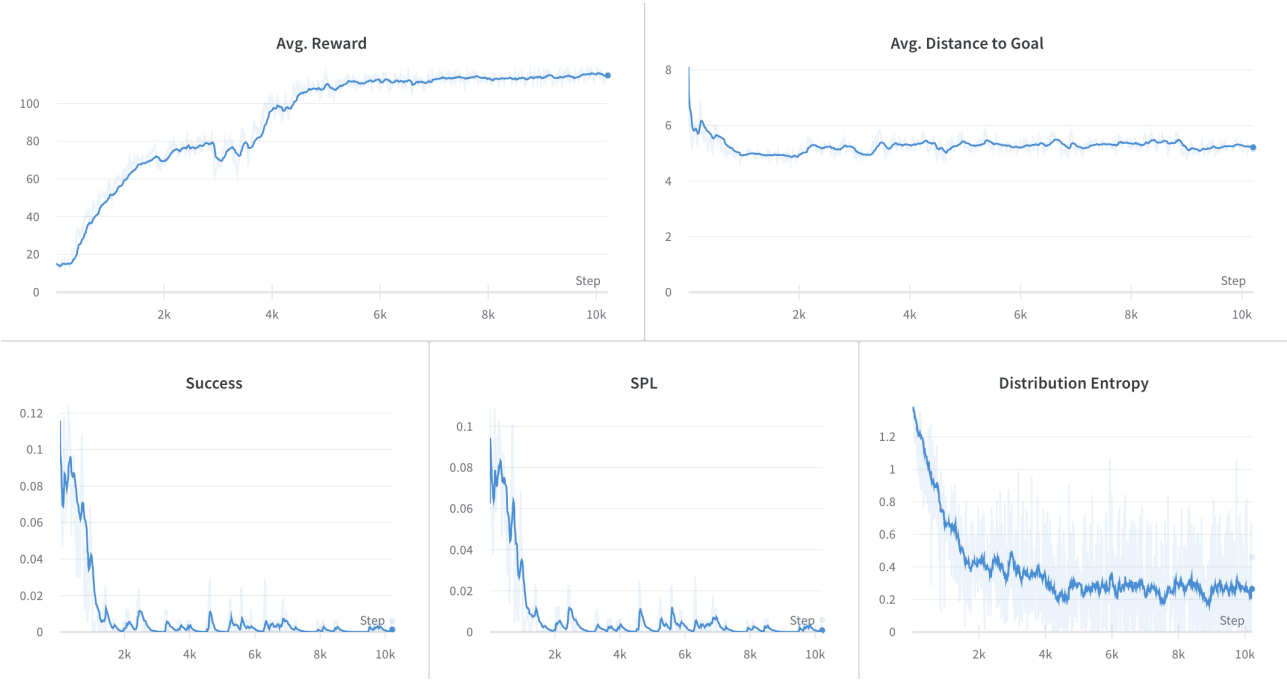


Figure 25: Summary of the SkillNav training.

Figure 26: Rewards of the evaluation of the 10 navigation skills in SkillNav.

goals. However, we also hypothesise that the evaluation of the agents is not consistent with the training outcomes, meaning that we are actually accomplishing the desired behaviours in the training phase. Also, the poor evaluation results can be due to only running an evaluation episode per skill, which defines a slightly biased evaluation.

6.3 Image Navigation

So far, we have defined and experimented with the *SkillNav* task for formally defining a set-up for learning the discovered skills. Additionally, we now propose experimentation with the *ImageNav* set-up. The latter is the baseline definition for tackling image goal-driven navigation in Habitat. Similar to SkillNav, ImageNav consists of an episodic navigation task where the objective is to reach a different goal in each episode. In ImageNav the goal is specified uniquely by an image. Compared to SkillNav, the reward that the agents

receive at each step is computed directly with respect to the actual distance in the environment between the agents and the goal. Formally, the differences and similarities between ImageNav and SkillNav are listed as follows.

- In ImageNav the agents start in a different location each episode while in SkillNav the start location is fixed to the center coordinate.
- In ImageNav there are approximately 70k different episodes (with different goals and initial locations) in each apartment while in SkillNav there are only 10 fixed goals which are self-discovered.
- In ImageNav, the task-oriented reward at each step is computed with the distance between the agent and the goal in the environment space, while in SkillNav the reward is computed in the latent space to maximize the MI (see Equation 5). Hence the reward in ImageNav is quite dense while in SkillNav is much more sparse.
- In ImageNav a goal position is characterized by an image that can be captured from that point with a random orientation while in SkillNav a goal position is characterized by a single embedding (either the centroid of the K-means clustering or a VQ-VAE codeword).
- In both ImageNav and SkillNav the goal is fed to the agents with an encoded embedding and later concatenated with the current observation embedding.
- In both ImageNav and SkillNav the agents have a fixed number of 150 steps to complete the episode.
- In both ImageNav and SkillNav the agents succeed in an episode if they call the *stop* action within a distance of 1 meter from the goal position.
- In both ImageNav and SkillNav we let the agents train for a total number of 3,000,000 steps.

The baseline approach for tackling ImageNav is very similar to the one that we use for SkillNav. The actions that the agents take are conditioned not only on the image state that they observe $\pi(a|s)$ but also on the conditioning goal $\pi(a|s, z)$. Regarding the baseline implementation provided in Habitat for tackling ImageNav, the network consists of a convolutional *goal visual encoder* and an identical *visual encoder*. The reason for working with the two encoders in parallel is not due to different architectures (because they are identical) but for updating their parameters separately both for agent observations and episode goals. On top of the two encoders the networks contain the PPO implementation that we use also for SkillNav.

In the experimentation with ImageNav we want to evaluate whether the representations that we learn in a task-agnostic fashion are useful for performing image goal-driven navigation. Concretely, we transfer the pre-trained contrastive-based main encoder with RGB data and train a PPO policy on top. Then, we compare the agents performance with the baseline on-line training of two convolutional encoders of the same size of the main encoder together with the policy on top of them.

In the white apartment we obtain the results in Figure 27. We see that pre-training the encoders provides useful capabilities to RL agents performing image goal-oriented navigation. Pre-training overcomes on-line training in the sense that it enables agents to obtain higher rewards, to be more successful in reaching the goals and to do so with shorter path lengths. Also, the entropy of the learn policy is lower in the pre-training case meaning that the learnt behaviours are more robust and consistent.

Figure 27: Summary of the ImageNav results in the white apartment. Comparison between (blue) pre-training the image encoders on the skill discovery phase and learning just the policy on top against (purple) learning both encoders and policy on-line.

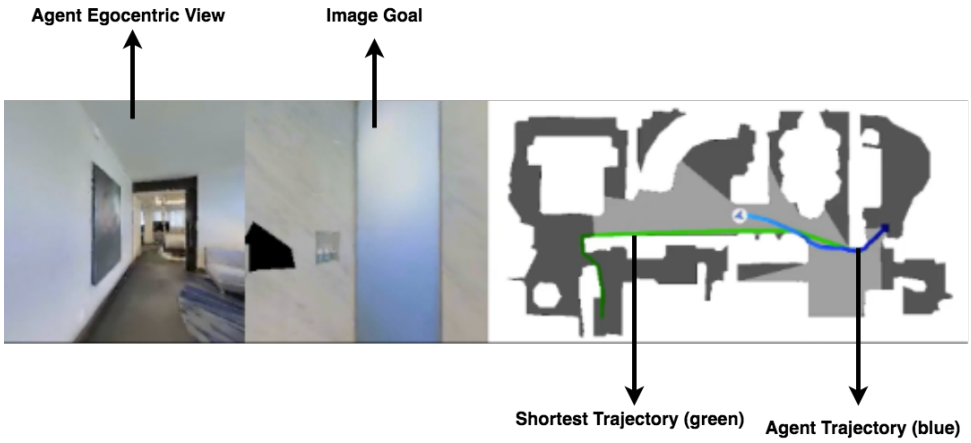
Figure 28: Summary of the ImageNav results in the circular apartment. Comparison between (blue) pre-training the image encoders on the skill discovery phase and learning just the policy on top against (purple) learning both encoders and policy on-line.

Additionally, we perform the same experimentation in both circular and maze apartments to assess if pre-training the encoders consistently overcomes on-line goal-driven navigation. We display the results obtained in the circular apartment in Figure 28.

Finally, Figure 29 shows the results in the maze apartment. In this case, there is no evidence that pre-training the encoders facilitates the downstream task. We hypothesise that transferring only the visual encoders might not be enough in the maze apartment due to the visual and dynamic complexity of the environment.

Figure 29: Summary of the ImageNav results in the maze apartment. Comparison between (blue) pre-training the image encoders on the skill discovery phase and learning just the policy on top against (purple) learning both encoders and policy on-line.

With this, we find that the learnt representations are generally useful for performing image-guided navigation even though we do not obtain very good results in SkillNav. We provide qualitative evaluation of the agents in this video[1]. The information that is displayed in the video follows the organization shown in Figure 6.3.



[1] https://youtu.be/WN4CGioUZ-k

7. Conclusions

We have extended the paradigm of Explore, Discover and Learn (EDL) to the pixel domain. We have also provided an end-to-end development and deployment pipeline of our methodology available in the Habitat AI environment [2]. We also provide the open-source repository of this project[2].

Regarding EDL, we have simplified the exploration task with non-parametric inference of the state distribution. However, we motivate that as EDL allows to tackle exploration independently of the posterior tasks, the former can be solved with more sophisticated techniques if needed (i.e. Random Network Distillation [34]).

Then, inspired from the information-theoretic objective of EDL, we have demonstrated the capabilities of variational inference to discover meaningful representations of a state space in the pixel domain and in an unsupervised manner. Furthermore, we have also leveraged a contrastive approach for the unsupervised discovery of the state representations from pixels. Also, we have demonstrated the robustness and consistency of the contrastive approach for obtaining meaningful representations in multiple environments and from different types of data. We hypothesize that it is the training pipeline design of the contrastive approach what allows the model to offer such consistent performance. We consider that the fact that we are capable of feeding as many different positive and negative image pairs as we want (because it is an easy process to craft both positive and negative examples in the way that we define them) allows the model to fit the image features distribution with high precision. In contrast, the reconstruction approach defines an embedded bottleneck and reconstruction loss that limits the model performance to its capability of extracting the most valuable features and representations from the images. In this way, we motivate that the implementation of more sophisticated encoder architectures in the VQ-VAE model would definitely increase the reconstruction approach performance.

Additionally, we have made use of the reverse form of the mutual information to derive a reward distribution for RL agents to provide a skill-learning framework. However, the results of the skill-learning stage are not as good as expected (specially in evaluation). We hypothesise that the problem relies in the evaluation of the agents since we obtain meaningful results in the training of the skill-learning stage. Also, we believe in our proposed methodology since we have externally demonstrated that it does work in other environments [63, 64].

Finally, we have demonstrated that pre-training the visual encoders of RL agents in a task-agnostic manner increases the agents performance in the task of image goal-oriented 3D navigation. With this, we motivate possible improvements and future work in the following stages.

- Usage of smarter exploratory policies in the exploration phase. We discuss that training policies to infer a uniform distribution over the state space or to intrinsically seek for novelty would provide consistency among the experimentation environments (thinking outside of Habitat AI simulator). Also, these more sophisticated techniques would relax the need of any prior information from the environment (in our implementation we retrieve a random navigable point to start each exploratory episode).
- Implementation of more sophisticated encoder architectures (i.e. ResNet encoders) for the self-supervised representation learning stage. Also consider parallel encoders for representing different

[2] `https://github.com/yuyecreus/Habitat-PixelEDL`

data types (RGB, depth, instance segmentation) and join the representations later in the process.

- Design of more dense reward distributions which maximization also coincides with the maximization of the reverse mutual information (i.e. use the distance or similarity in the embedding space and not just binary rewards).
- Evaluation of ImageNav between our approach and the baseline implementation in more apartments to consistently demonstrate that transferring the pre-trained encoders increases the agents performance.

Following the inspiration from EDL [1], this work has contributed to the research pipeline of *PixelEDL: Unsupervised Skill Discovery and Learning from Pixels*[3] [63] and *PiCoEDL: Discovery and Learning of Minecraft Navigation Goals from Pixels and Coordinates*[4] [64], which are peer-reviewed publications in the CVPR 2021 workshop on embodied artificial intelligence[5]. We have also submitted *Unsupervised Skill-Discovery and Skill-Learning in Minecraft* to the ICML 2021 workshop in Unsupervised Reinforcement Learning[6], and we are currently under review. One can find the three papers, the poster presentation of *PixelEDL: Unsupervised Skill Discovery and Learning from Pixels* and a screenshot of the presentation of our work in the CVPR 2021 workshop on embodied artificial intelligence in the Appendix section.

[3] https://imatge-upc.github.io/PixelEDL/
[4] https://imatge-upc.github.io/PiCoEDL/
[5] https://embodied-ai.org/
[6] https://urlworkshop.github.io/

References

[1] Víctor Campos, Alexander Trott, Caiming Xiong, Richard Socher, Xavier Giro-i Nieto, and Jordi Torres. Explore, discover and learn: Unsupervised discovery of state-covering skills. In *International Conference on Machine Learning*, pages 1317–1327. PMLR, 2020.

[2] Manolis Savva, Abhishek Kadian, Oleksandr Maksymets, Yili Zhao, Erik Wijmans, Bhavana Jain, Julian Straub, Jia Liu, Vladlen Koltun, Jitendra Malik, et al. Habitat: A platform for embodied ai research. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9339–9347, 2019.

[3] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995.

[4] Claude Sammut and Geoffrey I. Webb, editors. *Classification Tree*, pages 171–171. Springer US, Boston, MA, 2010.

[5] Claude Sammut and Geoffrey I. Webb, editors. *Logistic Regression*, pages 631–631. Springer US, Boston, MA, 2010.

[6] Arthur E Hoerl and Robert W Kennard. Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics*, 12(1):55–67, 1970.

[7] Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society: Series B (Methodological)*, 58(1):267–288, 1996.

[8] Claude Sammut and Geoffrey I. Webb, editors. *CART*, pages 147–147. Springer US, Boston, MA, 2010.

[9] James MacQueen et al. Some methods for classification and analysis of multivariate observations. In *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, volume 1, pages 281–297. Oakland, CA, USA, 1967.

[10] Fionn Murtagh and Pierre Legendre. Ward's hierarchical agglomerative clustering method: which algorithms implement ward's criterion? *Journal of classification*, 31(3):274–295, 2014.

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

[12] Gene H Golub and Christian Reinsch. Singular value decomposition and least squares solutions. In *Linear algebra*, pages 134–151. Springer, 1971.

[13] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[14] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014.

[15] Matthew Johnson, Katja Hofmann, Tim Hutton, and David Bignell. The malmo platform for artificial intelligence experimentation. In *IJCAI*, pages 4246–4247. Citeseer, 2016.

[16] Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. Openai gym. *arXiv preprint arXiv:1606.01540*, 2016.

[17] Lukasz Kaiser, Mohammad Babaeizadeh, Piotr Milos, Blazej Osinski, Roy H Campbell, Konrad Czechowski, Dumitru Erhan, Chelsea Finn, Piotr Kozakowski, Sergey Levine, et al. Model-based reinforcement learning for atari. *arXiv preprint arXiv:1903.00374*, 2019.

[18] Christopher Berner, Greg Brockman, Brooke Chan, Vicki Cheung, Przemyslaw Debiak, Christy Dennison, David Farhi, Quirin Fischer, Shariq Hashme, Chris Hesse, et al. Dota 2 with large scale deep reinforcement learning. *arXiv preprint arXiv:1912.06680*, 2019.

[19] Oriol Vinyals, Igor Babuschkin, Wojciech M Czarnecki, Michaël Mathieu, Andrew Dudzik, Junyoung Chung, David H Choi, Richard Powell, Timo Ewalds, Petko Georgiev, et al. Grandmaster level in starcraft ii using multi-agent reinforcement learning. *Nature*, 575(7782):350–354, 2019.

[20] Julian Schrittwieser, Ioannis Antonoglou, Thomas Hubert, Karen Simonyan, Laurent Sifre, Simon Schmitt, Arthur Guez, Edward Lockhart, Demis Hassabis, Thore Graepel, et al. Mastering atari, go, chess and shogi by planning with a learned model. *Nature*, 588(7839):604–609, 2020.

[21] Kun Shao, Zhentao Tang, Yuanheng Zhu, Nannan Li, and Dongbin Zhao. A survey of deep reinforcement learning in video games. *arXiv preprint arXiv:1912.10944*, 2019.

[22] Adam Daniel Laud. Theory and application of reward shaping in reinforcement learning. Technical report, 2004.

[23] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13):3521–3526, 2017.

[24] Yuri Burda, Harri Edwards, Deepak Pathak, Amos Storkey, Trevor Darrell, and Alexei A Efros. Large-scale study of curiosity-driven learning. *arXiv preprint arXiv:1808.04355*, 2018.

[25] Christoph Salge, Cornelius Glackin, and Daniel Polani. Empowerment–an introduction. In *Guided Self-Organization: Inception*, pages 67–114. Springer, 2014.

[26] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

[27] Ilge Akkaya, Marcin Andrychowicz, Maciek Chociej, Mateusz Litwin, Bob McGrew, Arthur Petron, Alex Paino, Matthias Plappert, Glenn Powell, Raphael Ribas, et al. Solving rubik's cube with a robot hand. *arXiv preprint arXiv:1910.07113*, 2019.

[28] Jemin Hwangbo, Inkyu Sa, Roland Siegwart, and Marco Hutter. Control of a quadrotor with reinforcement learning. *IEEE Robotics and Automation Letters*, 2(4):2096–2103, 2017.

[29] Adam Stooke, Kimin Lee, Pieter Abbeel, and Michael Laskin. Decoupling representation learning from reinforcement learning. *arXiv preprint arXiv:2009.08319*, 2020.

[30] Denis Yarats, Rob Fergus, Alessandro Lazaric, and Lerrel Pinto. Reinforcement learning with prototypical representations. *arXiv preprint arXiv:2102.11271*, 2021.

[31] Michael Laskin, Aravind Srinivas, and Pieter Abbeel. Curl: Contrastive unsupervised representations for reinforcement learning. In *International Conference on Machine Learning*, pages 5639–5650. PMLR, 2020.

[32] M Jaderberg, V Mnih, WM Czarnecki, T Schaul, JZ Leibo, D Silver, and K Kavukcuoglu. Reinforcement learning with unsupervised auxiliary tasks, in 'iclr'. 2017.

[33] Deepak Pathak, Pulkit Agrawal, Alexei A Efros, and Trevor Darrell. Curiosity-driven exploration by self-supervised prediction. In *International Conference on Machine Learning*, pages 2778–2787. PMLR, 2017.

[34] Yuri Burda, Harrison Edwards, Amos Storkey, and Oleg Klimov. Exploration by random network distillation. In *International Conference on Learning Representations*, 2018.

[35] Benjamin Eysenbach, Abhishek Gupta, Julian Ibarz, and Sergey Levine. Diversity is all you need: Learning skills without a reward function. In *International Conference on Learning Representations*, 2018.

[36] Karol Gregor, Danilo Jimenez Rezende, and Daan Wierstra. Variational intrinsic control. In *International Conference on Learning Representations (workshop)*, 2017.

[37] Xinlei Chen, Saining Xie, and Kaiming He. An empirical study of training self-supervised visual transformers. *arXiv preprint arXiv:2104.02057*, 2021.

[38] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. *arXiv preprint arXiv:2104.14294*, 2021.

[39] Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, 2020.

[40] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021.

[41] David Warde-Farley, Tom Van de Wiele, Tejas Kulkarni, Catalin Ionescu, Steven Hansen, and Volodymyr Mnih. Unsupervised control through non-parametric discriminative rewards. *arXiv preprint arXiv:1811.11359*, 2018.

[42] Aaron van den Oord, Oriol Vinyals, and Koray Kavukcuoglu. Neural discrete representation learning. *arXiv preprint arXiv:1711.00937*, 2017.

[43] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.

[44] Alex Lee, Anusha Nagabandi, Pieter Abbeel, and Sergey Levine. Stochastic latent actor-critic: Deep reinforcement learning with a latent variable model. *Advances in Neural Information Processing Systems*, 33, 2020.

[45] Víctor Campos, Pablo Sprechmann, Steven Hansen, Andre Barreto, Steven Kapturowski, Alex Vitvitskyi, Adrià Puigdomènech Badia, and Charles Blundell. Coverage as a principle for discovering transferable behavior in reinforcement learning. *arXiv preprint arXiv:2102.13515*, 2021.

[46] Elad Hazan, Sham Kakade, Karan Singh, and Abby Van Soest. Provably efficient maximum entropy exploration. In *International Conference on Machine Learning*, pages 2681–2691. PMLR, 2019.

[47] Navneet Madhu Kumar. Empowerment-driven exploration using mutual information estimation. *arXiv preprint arXiv:1810.05533*, 2018.

[48] Alexander S Klyubin, Daniel Polani, and Chrystopher L Nehaniv. All else being equal be empowered. In *European Conference on Artificial Life*, pages 744–753. Springer, 2005.

[49] Alexander S Klyubin, Daniel Polani, and Chrystopher L Nehaniv. Keep your options open: An information-based driving principle for sensorimotor systems. *PloS one*, 3(12):e4018, 2008.

[50] Joshua Achiam, Harrison Edwards, Dario Amodei, and Pieter Abbeel. Variational option discovery algorithms. *arXiv preprint arXiv:1807.10299*, 2018.

[51] Shakir Mohamed and Danilo Jimenez Rezende. Variational information maximisation for intrinsically motivated reinforcement learning. *arXiv preprint arXiv:1509.08731*, 2015.

[52] Guido Montúfar, Keyan Ghazi-Zahedi, and Nihat Ay. Information theoretically aided reinforcement learning for embodied agents. *arXiv preprint arXiv:1605.09735*, 2016.

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

[54] Angel Chang, Angela Dai, Thomas Funkhouser, Maciej Halber, Matthias Niessner, Manolis Savva, Shuran Song, Andy Zeng, and Yinda Zhang. Matterport3d: Learning from rgb-d data in indoor environments. *arXiv preprint arXiv:1709.06158*, 2017.

[55] Kuan Fang, Alexander Toshev, Li Fei-Fei, and Silvio Savarese. Scene memory transformer for embodied agents in long-horizon tasks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 538–547, 2019.

[56] Devendra Singh Chaplot, Dhiraj Prakashchand Gandhi, Abhinav Gupta, and Russ R Salakhutdinov. Object goal navigation using goal-oriented semantic exploration. *Advances in Neural Information Processing Systems*, 33, 2020.

[57] Dmytro Mishkin, Alexey Dosovitskiy, and Vladlen Koltun. Benchmarking classic and learned navigation in complex 3d environments. *arXiv preprint arXiv:1901.10915*, 2019.

[58] Hugh Durrant-Whyte and Tim Bailey. Simultaneous localization and mapping: part i. *IEEE robotics & automation magazine*, 13(2):99–110, 2006.

[59] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

[60] Yuke Zhu, Roozbeh Mottaghi, Eric Kolve, Joseph J Lim, Abhinav Gupta, Li Fei-Fei, and Ali Farhadi. Target-driven visual navigation in indoor scenes using deep reinforcement learning. In *2017 IEEE international conference on robotics and automation (ICRA)*, pages 3357–3364. IEEE, 2017.

[61] Tom Schaul, Daniel Horgan, Karol Gregor, and David Silver. Universal value function approximators. pages 1312–1320. PMLR, 2015.

[62] Adrian Łańcucki, Jan Chorowski, Guillaume Sanchez, Ricard Marxer, Nanxin Chen, Hans JGA Dolfing, Sameer Khurana, Tanel Alumäe, and Antoine Laurent. Robust training of vector quantized bottleneck models. In *2020 International Joint Conference on Neural Networks (IJCNN)*, pages 1–7. IEEE, 2020.

[63] Roger Creus Castanyer, Juan José Nieto, and Xavier Giro-i Nieto. Pixeledl: Unsupervised skill discovery and learning from pixels.

[64] Juan José Nieto, Roger Creus Castanyer, and Xavier Giro-i Nieto. Picoedl: Discovery and learning of minecraft navigation goals from pixels and coordinates.

Appendices

VQ-VAE	
learning rate	1e-3
batch size	256
epochs	300
num hiddens	64
num residual hiddens	32
num residual layers	2
embedding dim	256
num embeddings	10
β	0.25
decay	0.99
k_μ	15
k_σ	5

ATC	
learning rate	1e-3
batch size	128
epochs	300
τ	5e-3
soft update	2
embedding dim	128
k_μ	15
k_σ	5

PPO	
learning rate	2.5e-4
number of minibatches	4
PPO epochs	4
RNN hidden size	128
number of steps	128
max episode steps	150
clip parameter	0.2
ADAM epsilon	1e-5
max gradient norm	0.5
gamma	0.99
tau	0.95

PixelEDL: Unsupervised Skill Discovery and Learning from Pixels

Roger Creus Castanyer[1] Juan José Nieto[1] Xavier Giro-i-Nieto[1,2,3]
[1]*Universitat Politècnica de Catalunya* [2]*Barcelona Supercomputing Center* [3]*Institut de Robòtica i Informàtica Industrial, CSIC-UPC*
creus99@protonmail.com juanjo.3ns@gmail.com xavier.giro@upc.edu

1. Introduction

We tackle embodied visual navigation in a task-agnostic set-up by putting the focus on the unsupervised discovery of skills (or options [2]) that provide a good coverage of states. Our approach intersects with empowerment [10]: we address the reward-free skill discovery and learning tasks to discover *what* can be done in an environment and *how*. For this reason, we adopt the existing Explore, Discover and Learn (EDL) [1] paradigm, tested only in toy example mazes, and extend it to pixel-based state representations available for embodied AI agents. The information-theoretic paradigm of EDL [1] aims to learn latent-conditioned policies, namely skills $\pi(a|s,z)$, by maximizing the mutual information (MI) between the inputs s and some latent variables z. Hence, EDL [1] consists of unsupervised skill discovery and training of reinforcement learning (RL) agents without considering the existence of any extrinsic motivation or reward. We present *PixelEDL*, an implementation of the EDL paradigm for the pixel representations provided by the AI Habitat [11] and MineRL [3] environments. In comparison with EDL, PixelEDL involves self-supervised representation learning of image observations for information-theoretic skill discovery. Still, PixelEDL aims to maximize the MI between inputs and some latent variables and for that it consists of the same three stages of EDL (explore, discover and learn). By breaking down the RL end-to-end training pipeline into the three stages, we also simplify the implicit difficulty in learning both representations and policies from a high dimensional input space all at once [7].

Figure 1. Top-down views of the three considered environments: (i) a custom "toy example" Minecraft map, (ii) a Realistic Minecraft map, and (iii) a Habitat apartment.

The results presented in this extended abstract are further extended in our project site[1].

2. Methodology

We assume an underlying Markov Decision Process (MDP) without rewards: $\mathcal{M} = (\mathcal{S}, \mathcal{A}, \mathcal{P})$ where $\mathcal{S}$ is the high-dimensional set of states (defined by image pixels). $\mathcal{A}$ is the action space and $\mathcal{P} = p(s|s,a)$ is the transition function. Moreover, we define the objective of PixelEDL as the maximization of the MI in equation (1), which requires knowledge of the unknown distributions $p(s), p(s|z), p(z|s)$.

$$\begin{aligned} \mathcal{I}(S,Z) &= \mathcal{H}(Z) - \mathcal{H}(Z|S) \quad \rightarrow \quad \text{reverse} \\ &= \mathcal{H}(S) - \mathcal{H}(S|Z) \quad \rightarrow \quad \text{forward} \end{aligned} \tag{1}$$

2.1. Exploration

The first task to tackle in PixelEDL is exploration. Without any prior knowledge, a reasonable choice for discovering state-covering skills is to define the distribution over all states $p(s)$ uniformly. However, training an exploration policy to infer a uniform $p(s)$ is not feasible in PixelEDL since it deals with the high-dimensional pixel space. To overcome this limitation we adopt a non-parametric estimation of $p(s)$ by sampling from a dataset of collected experience. Hence, in PixelEDL the goal of the exploration stage is to collect a dataset of trajectories containing representative states that the learned skills should ultimately cover. PixelEDL adopts a random exploration of the environment through agents that perform random actions within a discrete action space (i.e. move forward, turn left, turn right) and collect the trajectories generated by the environment. For our custom Minecraft map, random policies from agents instantiated in the center of the map are capable of covering a complete set of representative states of the environment given a large number of episodes. In the realistic Minecraft map the random agents do not cover as many representative states as in the custom map but still provide enough coverage of the state space. However, in order to obtain a set

[1] https://imatge-upc.github.io/PixelEDL/

of representative states of the Habitat map we let the agents start in random navigable points of the map at each episode.

2.2. Skill Discovery

The Discovery stage of EDL aims at finding the latent representations z that will ultimately condition the agent policies to learn the skills $\pi(a|s, z)$. Hence, the goal of PixelEDL in this stage is to model $p(z|s)$ as a mapping of the states to their representations and to model $p(z)$ as a categorical distribution of meaningful representations.

Ideally, we aim to obtain representations of the image observations that encode existing similarities and spatial relations within the environment [6]. Furthermore, we aim to find z that are representatives of a meaningful segmentation of the state space. In this work the representations z will be later used to condition a navigation task. Previous works [16] have reported the challenges of unsupervised learning of representations from images that encode valuable features for RL agents in a 3D environment. For modelling $p(z|s)$, we study the performance of two different approaches: (i) a *contrastive* one, that uses a siamese architecture and aims to project positive pairs of input images closer in an embedding space, and (ii) a *reconstruction* one, that use a Variational Autoencoder (VAE) [5] with categorical classes, namely Vector Quantisation VAE (VQ-VAE) [9], to train the model to reconstruct the observations. For the contrastive approach, we use the adaptation to CURL [14] proposed by Stooke et al. [15], namely Augmented Temporal Contrast (ATC). Compared to CURL, in ATC the positive pairs of inputs consist of two image observations belonging to the same exploration trajectory. That is, we train both ATC and VQ-VAE so that a positive pair of inputs consists of two observations of the same trajectory with a delay $d \sim \mathcal{N}(\mu, \sigma^2)$. We experiment with $\mu = 15$ and $\sigma = 5$. Hence, in both ATC and VQ-VAE we perform a data augmentation in the temporal domain. Our experiments indicate that the capabilities of both ATC and VQ-VAE for modelling $p(z|s)$ are promising and we have not yet observed important differences to justify using one over the other.

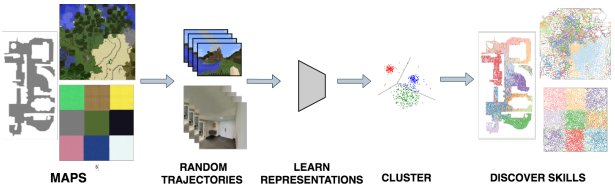


Figure 2. Self-Supervised representation learning and unsupervised skill discovery pipeline.

After the visual representation learning, we model a categorical distribuiton $p(z)$ by clustering the embedding space of the representations. Yarats et. al [17] use a projection of the embeddings onto the prototypes which define a basis of the embedding space to perform the cluster assignments. However, in VQ-VAE, this clustering is implicit in the model since the cluster centroids are actually the representatives of the model's codebook. Also, for ATC we apply a K-means [8] clustering for modelling $p(z)$ with the cluster centroids. After modelling $p(z)$ and $p(z|s)$, we complete the stages of representation learning and skill discovery. Figure 2 summarises the aforementioned pipeline. We provide more details in our project site.

2.3. Learning

Given a model of $p(z)$, we make use of the formulation of Universal Value Function Approximators (UVFA) [12] to train a policy to maximize the MI (1) between the inputs and z. That is, we exploit z as navigation goals or intrinsic objectives to learn the goal-conditioned skills: $\pi(a|s, z)$. Hence, we feed the concatenation of the encoded observation and z to the RL agents. Thus, at each step, the policy predictions depend not only on the current agent state but also on z. EDL [1] maximizes the forward form of the MI (1). That is feasible in EDL because the technique is applied to toy mazes where the states of the MDP are defined by 2D coordinates. In this way, EDL models $p(s|z)$ by variational inference and maximize the MI by deriving a reward that involves computing euclidean distances in the state space of coordinates. However, as in PixelEDL we deal with the pixel space, it is not coherent to match the euclidean distance in the image observation space with the distances in the 3D environment. For this reason we make use of the reverse form of the MI (1) and we model $p(z|s)$ with the encoder that learns latent representations from image observations. Finally, we craft a reward distribution that maximizes the MI (1) between the inputs and the skills by taking into account the distances in the latent space of the representations. Concretely, we assign a positive reward to an action a that positions the agents in a state s only if the encoded image observation is closest to the skill-conditioning z among all $z \sim p(z)$. We use the baseline RL models provided by both Habitat and MineRL for implementing the aforementioned training pipeline. These models are Proximal Policy Optimization (PPO) [13] and Rainbow [4] respectively.

While PixelEDL is capable of learning some of the discovered skills, specially in the custom Minecraft map, it finds more difficulties in the realistic one and in Habitat. We hypothesize that: (i) Rainbow struggles with the latent codes z that encode similar regions of the realistic Minecraft state space; (ii) further tuning of PPO could achieve better results when learning the skills in Habitat, since we already obtain high-quality discovery of these. We provide qualitative results together with a demo video in our project site.

References

[1] Víctor Campos, Alexander Trott, Caiming Xiong, Richard Socher, Xavier Giro-i Nieto, and Jordi Torres. Explore, discover and learn: Unsupervised discovery of state-covering skills. In *International Conference on Machine Learning*, pages 1317–1327. PMLR, 2020. 1, 2

[2] Karol Gregor, Danilo Jimenez Rezende, and Daan Wierstra. Variational intrinsic control. *arXiv preprint arXiv:1611.07507*, 2016. 1

[3] William H. Guss, Mario Ynocente Castro*, Sam Devlin*, Brandon Houghton*, Noboru Sean Kuno*, Crissman Loomis*, Stephanie Milani*, Sharada Mohanty*, Keisuke Nakata*, Ruslan Salakhutdinov*, John Schulman*, Shinya Shiroshita*, Nicholay Topin*, Avinash Ummadisingu*, and Oriol Vinyals*. Neurips 2020 competition: The MineRL competition on sample efficient reinforcement learning using human priors. *NeurIPS Competition Track*, 2020. 1

[4] Matteo Hessel, Joseph Modayil, Hado Van Hasselt, Tom Schaul, Georg Ostrovski, Will Dabney, Dan Horgan, Bilal Piot, Mohammad Azar, and David Silver. Rainbow: Combining improvements in deep reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018. 2

[5] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013. 2

[6] Sascha Lange and Martin Riedmiller. Deep auto-encoder neural networks in reinforcement learning. In *The 2010 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8, 2010. 2

[7] Alex X Lee, Anusha Nagabandi, Pieter Abbeel, and Sergey Levine. Stochastic latent actor-critic: Deep reinforcement learning with a latent variable model. *arXiv preprint arXiv:1907.00953*, 2019. 1

[8] James MacQueen et al. Some methods for classification and analysis of multivariate observations. In *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, volume 1, pages 281–297. Oakland, CA, USA, 1967. 2

[9] Aaron van den Oord, Oriol Vinyals, and Koray Kavukcuoglu. Neural discrete representation learning. *arXiv preprint arXiv:1711.00937*, 2017. 2

[10] Christoph Salge, Cornelius Glackin, and Daniel Polani. Empowerment–an introduction. In *Guided Self-Organization: Inception*, pages 67–114. Springer, 2014. 1

[11] Manolis Savva, Abhishek Kadian, Oleksandr Maksymets, Yili Zhao, Erik Wijmans, Bhavana Jain, Julian Straub, Jia Liu, Vladlen Koltun, Jitendra Malik, et al. Habitat: A platform for embodied ai research. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9339–9347, 2019. 1

[12] Tom Schaul, Daniel Horgan, Karol Gregor, and David Silver. Universal value function approximators. In *International conference on machine learning*, pages 1312–1320. PMLR, 2015. 2

[13] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017. 2

[14] Aravind Srinivas, Michael Laskin, and Pieter Abbeel. Curl: Contrastive unsupervised representations for reinforcement learning. *arXiv preprint arXiv:2004.04136*, 2020. 2

[15] Adam Stooke, Kimin Lee, Pieter Abbeel, and Michael Laskin. Decoupling representation learning from reinforcement learning. *arXiv preprint arXiv:2009.08319*, 2020. 2

[16] David Warde-Farley, Tom Van de Wiele, Tejas Kulkarni, Catalin Ionescu, Steven Hansen, and Volodymyr Mnih. Unsupervised control through non-parametric discriminative rewards. *arXiv preprint arXiv:1811.11359*, 2018. 2

[17] Denis Yarats, Rob Fergus, Alessandro Lazaric, and Lerrel Pinto. Reinforcement learning with prototypical representations. *arXiv preprint arXiv:2102.11271*, 2021. 2

PiCoEDL: Discovery and Learning of Minecraft Navigation Goals from Pixels and Coordinates

Juan José Nieto[1] Roger Creus Castanyer[1] Xavier Giro-i-Nieto[1,2,3]
[1]*Universitat Politècnica de Catalunya* [2]*Barcelona Supercomputing Center* [3]*Institut de Robòtica i Informàtica Industrial, CSIC-UPC*
juanjo.3ns@gmail.com creus99@protonmail.com xavier.giro@upc.edu

1. Introduction

Defining a reward function in Reinforcement Learning (RL) is not always possible or very costly. For this reason, there is a great interest in training agents in a task-agnostic manner making use of intrinsic motivations and unsupervised techniques [7, 6, 15, 2, 14, 3]. Due to the complexity to learn useful behaviours in pixel-based domains, the results obtained in RL are still far from the remarkable results obtained in domains such as computer vision [5, 4] or natural language processing [1, 12]. We hypothesize that RL agents will also benefit from unsupervised pre-trainings with no extrinsic rewards, analogously to how humans mostly learn, especially in the early stages of life.

Our main contribution is the deployment of the *Explore, Discover and Learn* (EDL) [3] paradigm for unsupervised learning to the pixel and coordinate space (PiCoEDL). In particular, our work focuses on the MineRL [9] environment, where the observation of the agent is represented by: (a) its spatial coordinates in the Minecraft virtual world, and (b) an image from an egocentric viewpoint. Following the idea of *empowerment* [10], our goal is to learn latent-conditioned policies by maximizing the mutual information between states and some latent variables, instead of sequences of actions [7]. This allows the agent to influence the environment while discovering available skills.

2. From pixels and coordinates to skills

We formulate a Markov decision process (MDP) as $\mathcal{M} = (\mathcal{S}, \mathcal{A}, \mathcal{P})$. $\mathcal{S}$ is the high-dimensional state space (pixel images and coordinates), $\mathcal{A}$ refers to the set of actions available in the environment and $\mathcal{P}$ defines the transition probability $p(s_{t+1}|s_t, a)$. We learn latent-conditioned policies $\pi(a|s, z)$, where the latent $z \in \mathcal{Z}$ is a random variable.

Given the property of symmetry, the mutual information ($\mathcal{I}$) can be written using the Shannon Entropy ($\mathcal{H}$) in two ways:

$$\begin{aligned} \mathcal{I}(S, Z) &= \mathcal{H}(Z) - \mathcal{H}(Z|S) \quad \rightarrow \quad \text{reverse} \\ &= \mathcal{H}(S) - \mathcal{H}(S|Z) \quad \rightarrow \quad \text{forward} \end{aligned} \tag{1}$$

Maximizing the mutual information (MI) requires knowledge of unknown distributions ($p(s), p(s|z), p(z|s)$). For the former we study two cases: (a) Using expert trajectories, and (b) Using the distribution induced by a random policy. A comparison of both strategies can be found in Section 2.1, for now we assume the latter case of $p(s)$. We rely on variational inference techniques for estimating the mappings from the states to the latent variables and backwards. These are estimated from the rollouts induced by the random policy. Finally, we also need to define a prior $p(z)$ to sample from, which in our case, following EDL [3] is a fixed uniform categorical distribution.

If we proceed with the derivation of the forward form in EDL [3], we can find that in our case the intrinsic objective becomes a distance in the pixel space. Since we cannot assume that it is representative of a meaningful distance in the environment, we discard this approach. Instead, we adopt the reverse form of the MI. We use the VQVAE [13] model, that allows us to estimate the posterior $p(z|s)$ with the encoder $q_\phi(z|s)$ by maximum likelihood on (s, z) tuples and also contains a categorical bottleneck for $p(z)$. Then, our final objective becomes:

$$r(s, z) = q_\phi(z = k|s) = \begin{cases} 1, & \text{if } k = \text{argmin}_j ||z_e(s) - e_j|| \\ 0, & \text{otherwise} \end{cases} \tag{2}$$

$z_e(s)$ is the sum of the outputs of two encoders: (a) a 2D convolutional encoder for the images, and (b) a multilayer perceptron for the coordinates. Both have the same output dimension that allows summing up the resulting embeddings. Then in Equation 2, we find the index of the closest embedding in the VQVAE codebook e. Only if this index matches the sampled latent variable z that is conditioning the policy, it will return a reward of 1. Despite the sparsity of rewards, since we use a $p(s)$ that is induced from a random policy, we

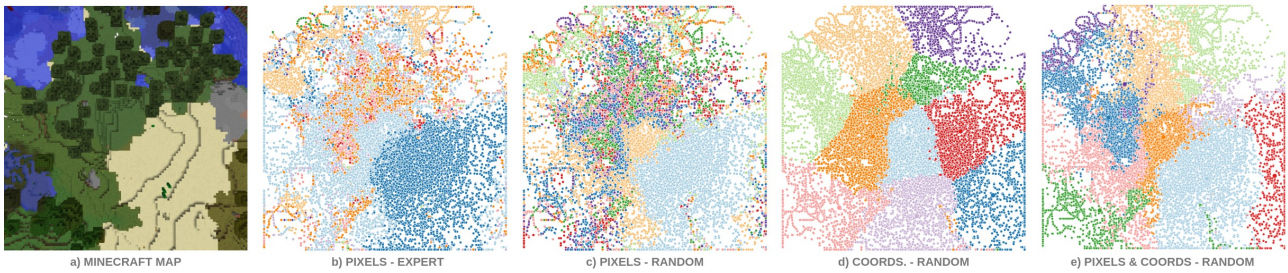


Figure 1. a) Top-view of our Minecraft map. The caption below b) c) d) e) refers to the nature of the states (pixels, coordinates or both) and the type of trajectories (expert or random). Each coloured point indicates the closest centroid to the encoded embedding.

know that these states are reachable and we will not suffer from exploration problems. Also, there are other problems due to multiple states rising positive rewards which could lead to ambiguous objectives. This can be tackled using a smoother reward function such as computing the distance in the embedding state, but in our experiments we did not have any problem by leveraging the previous reward.

In the following subsections we specify the implementation details of our approach.

2.1. Exploration and Skill Discovery

Firstly, we considered using expert trajectories to induce the distribution over the states. We used the MineRL dataset [8], which contains expert trajectories from different Minecraft worlds. Using expert trajectories may seem preferable since they contain human priors that give more weight to those states that are meaningful for discovering useful skills. However, Figure 1b shows that expert trajectories from pixels discover fewer and sparser skills than the those discovered by random exploration in our map, depicted in Figure 1c. This suggests that while the skills discovered by experts may be more generic as they were collected in different worlds, they are not as useful for our particular map as the skills discovered by a random policy. This hypothesis is supported by Figure 2, where we show the reconstructed images for each of the VQ-VAE codebook centroids. The reconstructions belonging to the expert trajectories contain scenarios that cannot be found in our Minecraft map.

In our study case, we aim to learn policies that treat the latent variables as navigation-goals. For this purpose, Figure 1 shows complementary skills discovered from pixels (Figure 1c) or coordinates (Figure 1d). We would like to discover skills that take into account not only the visual similarity but also the position relative to the initial state, so we adopt a solution that considers the two types of state representations (Figure 1e). This way our agent can distinguish between two visually identical mountains located at two different positions in the map.

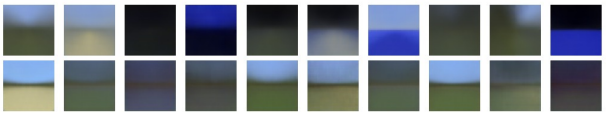


Figure 2. **Top:** images reconstructed from learned centroids using expert trajectories. **Bottom:** images reconstructed from learned centroids using pixels and coordinates from random trajectories.

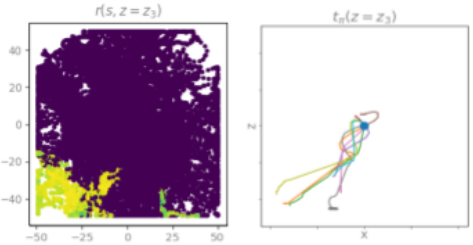


Figure 3. **Left:** Observations in yellow are encoded to the codebook embeddding z_3. **Right:** Trajectories followed by the agent conditioned by z_3.

2.2. Skill-Learning

In the last stage of PiCoEDL, we leverage Equation 2, derived from maximizing the mutual information between states and latent variables to maximize the expected cumulative reward. The latent codes discovered are now treated as goal states in a navigation task. We utilize Rainbow [11] algorithm to train our RL embodied agent. The input to the network is composed of the concatenation of the embedded observation with the latent embedding that is conditioning the policy. For each episode, we sample uniformly from $p(z)$ to determine the conditioning latent.

While there are some policies that are correctly learned, we find some others that do not achieve satisfactory results. We hypothesize that these are the latent codes that encode smaller regions of the state space, and with further tuning may achieve the desirable results. Figure 3 depicts the trained policy conditioned with the third codebook. More examples are available in our project site[1].

[1] https://imatge-upc.github.io/PiCoEDL/

References

[1] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, et al. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*, 2020. 1

[2] Y. Burda, H. Edwards, A. Storkey, and O. Klimov. Exploration by random network distillation. *arXiv preprint arXiv:1810.12894*, 2018. 1

[3] V. Campos, A. Trott, C. Xiong, R. Socher, X. Giro-i Nieto, and J. Torres. Explore, discover and learn: Unsupervised discovery of state-covering skills. In *International Conference on Machine Learning*, pages 1317–1327. PMLR, 2020. 1

[4] M. Caron, H. Touvron, I. Misra, H. Jégou, J. Mairal, P. Bojanowski, and A. Joulin. Emerging properties in self-supervised vision transformers, 2021. 1

[5] X. Chen, S. Xie, and K. He. An empirical study of training self-supervised visual transformers. *arXiv preprint arXiv:2104.02057*, 2021. 1

[6] B. Eysenbach, A. Gupta, J. Ibarz, and S. Levine. Diversity is all you need: Learning skills without a reward function. *arXiv preprint arXiv:1802.06070*, 2018. 1

[7] K. Gregor, D. J. Rezende, and D. Wierstra. Variational intrinsic control. *arXiv preprint arXiv:1611.07507*, 2016. 1

[8] W. H. Guss, B. Houghton, N. Topin, P. Wang, C. Codel, M. Veloso, and R. Salakhutdinov. Minerl: A large-scale dataset of minecraft demonstrations. In *IJCAI*, 2019. 2

[9] W. H. Guss, M. Y. Castro, S. Devlin, B. Houghton, N. S. Kuno, C. Loomis, S. Milani, S. Mohanty, K. Nakata, R. Salakhutdinov, et al. The minerl 2020 competition on sample efficient reinforcement learning using human priors. *arXiv preprint arXiv:2101.11071*, 2021. 1

[10] D. Hafner, P. A. Ortega, J. Ba, T. Parr, K. Friston, and N. Heess. Action and perception as divergence minimization. *arXiv preprint arXiv:2009.01791*, 2020. 1

[11] M. Hessel, J. Modayil, H. Van Hasselt, T. Schaul, G. Ostrovski, W. Dabney, D. Horgan, B. Piot, M. Azar, and D. Silver. Rainbow: Combining improvements in deep reinforcement learning. In *AAAI*, 2018. 2

[12] Z. Lan, M. Chen, S. Goodman, K. Gimpel, P. Sharma, and R. Soricut. Albert: A lite bert for self-supervised learning of language representations. *arXiv preprint arXiv:1909.11942*, 2019. 1

[13] A. v. d. Oord, O. Vinyals, and K. Kavukcuoglu. Neural discrete representation learning. *arXiv preprint arXiv:1711.00937*, 2017. 1

[14] D. Pathak, P. Agrawal, A. A. Efros, and T. Darrell. Curiosity-driven exploration by self-supervised prediction. In *International Conference on Machine Learning*, pages 2778–2787. PMLR, 2017. 1

[15] D. Warde-Farley, T. Van de Wiele, T. Kulkarni, C. Ionescu, S. Hansen, and V. Mnih. Unsupervised control through non-parametric discriminative rewards. *arXiv preprint arXiv:1811.11359*, 2018. 1

UNIVERSITAT POLITÈCNICA DE CATALUNYA BARCELONATECH
Institut de Robòtica i Informàtica Industrial
Barcelona Supercomputing Center Centro Nacional de Supercomputación

PixelEDL: Unsupervised Skill Discovery and Learning from Pixels


Roger Creus Castanyer

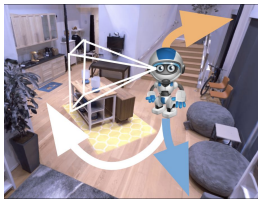
Juan José Nieto

Xavier Giro-i-Nieto

CVPR VIRTUAL JUNE 19-25

Task description

Approach task-agnostic visual embodied navigation by **discovering navigation goals** and **learning behaviours** as conditioned policies to accomplish the goals through an intrinsic motivation **(empowerment)**.

3D Minecraft and Habitat environment. Input consists of RGB observations only.

Motivation

EMPOWERMENT

Increase **agents influence** over the environment

SKILL DISCOVERY

Learn meaningful **task-agnostic state representations**

INFORMATION THEORETIC OBJECTIVE

Maximize **mutual information**

$$I(S;Z) = H(S) - H(S|Z)$$
$$= H(Z) - H(Z|S)$$

Challenges

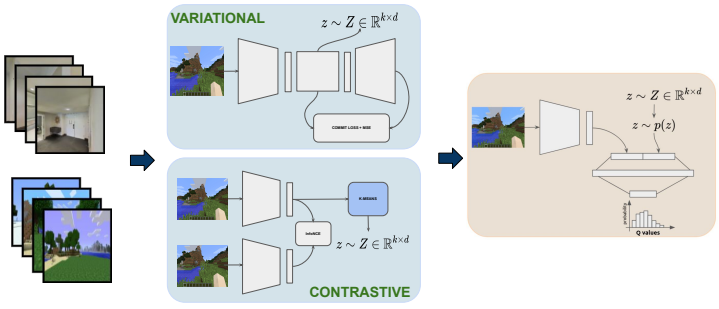
1. **Intrinsic reward** derived from the **forward form** of the Mutual Information **does not match a distance** in an environment characterized by **image observations**.
2. Discover disjoint navigation goals by taking into account **image observation resemblance only.**
3. Scale from **visually simple to complex** environments

CUSTOM MINECRAFT MAP | REALISTIC MINECRAFT MAP | HABITAT APARTMENT

Our approach

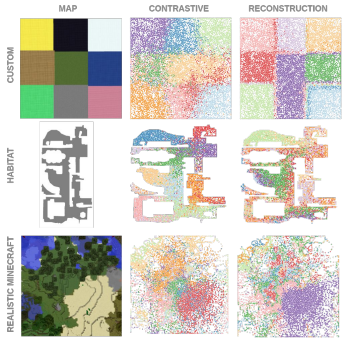
Based on **EDL** (Explore, Discover and Learn) by **Víctor Campos** et. al.

EXPLORE | **DISCOVER** | **LEARN**

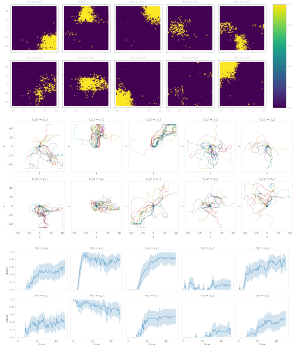


Results

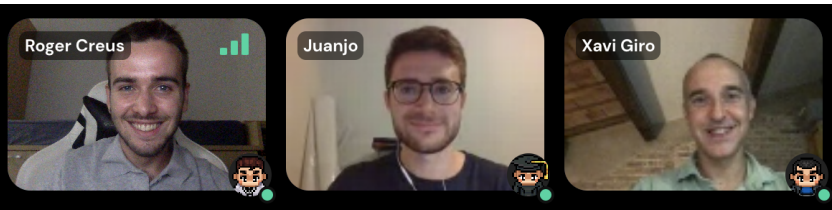
SKILL DISCOVERY



SKILL LEARNING



https://imatge-upc.github.io/PixelEDL/ https://imatge-upc.github.io/PiCoEDL/



PixelEDL: Unsupervised Skill Discovery and Learning from Pixels

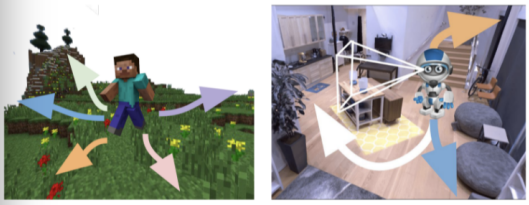
UNIVERSITAT POLITÈCNICA DE CATALUNYA BARCELONATECH
Institut de Robòtica i Informàtica Industrial
Barcelona Supercomputing Center
Centro Nacional de Supercomputación

Roger Creus Castanyer
Juan José Nieto
Xavier Giro-i-Nieto

CVPR VIRTUAL JUNE 19-25
CVPR 2021 EMBODIED AI

Task description

Approach task-agnostic visual embodied navigation by **discovering navigation goals** and **learning behaviours** as conditioned policies to accomplish the goals through an intrinsic motivation **(empowerment)**.



3D Minecraft and Habitat environment. Input consists of RGB observations only.

Motivation

EMPOWERMENT

Maximize **agents influence** over the environment

SKILL DISCOVERY

Learn meaningful **task-agnostic state representations**

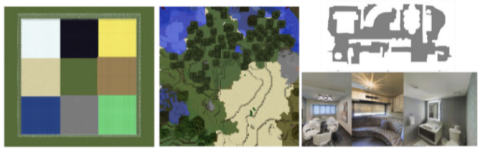
INFORMATION THEORETIC OBJECTIVE

Maximize **mutual information**

$$I(S;Z) = H(S) - H(S|Z)$$
$$= H(Z) - H(Z|S)$$

Challenges

1. **Intrinsic reward** derived from the **forward form** of the Mutual Information **does not match a distance** in an environment characterized by **image observations**.
2. Discover disjoint navigation goals by taking into account **image observation resemblance only.**
3. Scale from **visually simple to complex** environments



CUSTOM MINECRAFT MAP | REALISTIC MINECRAFT MAP | HABITAT APARTMENT

Our approach

Based on **EDL [1]** (Explore, Discover and Learn) by **Víctor Campos** et. al.

Results

SKILL DISCOVERY | SKILL LEARNING

Unsupervised Skill-Discovery and Skill-Learning in Minecraft

Juan José Nieto [1] **Roger Creus** [1] **Xavier Giro-i-Nieto** [1 2 3]

Abstract

Pre-training Reinforcement Learning agents in a task-agnostic manner has shown promising results. However, previous works still struggle in learning and discovering meaningful skills in high-dimensional state-spaces, such as pixel-spaces. We approach the problem by leveraging unsupervised skill discovery and self-supervised learning of state representations. In our work, we learn a compact latent representation by making use of variational and contrastive techniques. We demonstrate that both enable RL agents to learn a set of basic navigation skills by maximizing an information theoretic objective. We assess our method in Minecraft 3D pixel maps with different complexities. Our results show that representations and conditioned policies learned from pixels are enough for toy examples, but do not scale to realistic and complex maps. To overcome these limitations, we explore alternative input observations such as the relative position of the agent along with the raw pixels.

1. Introduction

Reinforcement Learning (RL) (Sutton and Barto, 2018) has witnessed a wave of outstanding works in the last decade, with special focus on games (Schrittwieser et al. (2020), Vinyals et al. (2019), Berner et al. (2019)), but also in robotics (Akkaya et al. (2019), Hwangbo et al. (2017)). In general, these works follow the classic RL paradigm where an agent interacts with an environment performing some action, and in response it receives a reward. These agents are optimized to maximize the expected sum of future rewards.

Rewards are usually handcrafted or overparametrized, and this fact becomes a bottleneck that prevents RL to scale. For this reason, there has been an increasing interest in training agents in a task-agnostic manner during the last few years, making use of intrinsic motivations and unsupervised techniques. Recent works have explored the unsupervised learning paradigm (Campos et al. (2020); Gregor et al. (2017); Eysenbach et al. (2018); Warde-Farley et al. (2018); Burda et al. (2018); Pathak et al. (2017)), but RL is still far from the remarkable results obtained in other domains. For instance, in computer vision, Chen et al. (2021) achieve an 81% accuracy on ImageNet training in a self-supervised manner, or Caron et al. (2021) achieves state-of-the-art results in image and video object segmentation using Visual Transformers (Dosovitskiy et al., 2021) and no labels at all. Also, in natural language processing pre-trained language models such as GPT-3 (Brown et al., 2020) have become the basis for other downstream tasks.

Humans and animals are sometimes guided through the process of learning. We have good priors that allow us to properly explore our surroundings, which leads to discovering new skills. For machines, learning skills in a task-agnostic manner has proved to be challenging (Warde-Farley et al., 2018; Lee et al., 2020). These works state that training pixel-based RL agents end-to-end is not efficient because learning a good state representation is unfeasible due to the high dimensionality of the observations. Moreover, most of the successes in RL come from training agents during thousands of simulated years (Berner et al. (2019)) or millions of games (Vinyals et al. (2019)). This learning approach is very sample inefficient and sometimes limits its research because of the high computational budget it may imply. As a response, some benchmarks have been proposed to promote the development of algorithms that can reduce the number of samples needed to solve complex tasks. This is the case of MineRL (Guss et al. (2021)) or ProcGen Benchmark (Cobbe et al. (2020)).

Our work is inspired by Campos et al. (2020) and their *Explore, Discover and Learn* (EDL) paradigm. EDL relies on *empowerment* (Salge et al., 2014) for motivating an agent intrinsically. Empowerment aims to maximize the influence of the agent over the environment while discovering novel skills. As stated by Salge et al. (2014), this can be achieved my maximizing the mutual information between sequences of actions and final states. Gregor et al. (2017) introduces

[1]Universitat Politècnica de Catalunya, Barcelona, Spain [2]Barcelona Supercomputing Center, Barcelona, Spain [3]Institut de Robòtica i Informàtica Industrial, CSIC-UPC, Barcelona, Spain. Correspondence to: Juan José Nieto <juanjo.3ns@gmail.com>, Roger Creus <creus99@protonmail.com>, Xavier Giro-i-Nieto <xavier.giro@upc.edu>.

Proceedings of the 38^{th} International Conference on Machine Learning, PMLR 139, 2021.

a novel approach that instead of blindly committing to a sequence of actions, each action depends on the observation from the environment. This is achieved by maximizing the mutual information between inputs and some latent variables. Campos et al. (2020) embraces this approach as we do. However, the implementation by Campos et al. (2020) makes some assumptions that are not realistic for pixel observations. Due to the Gaussian assumption at the output of variational approaches, the intrinsic reward is computed as the reconstruction error, and in the pixel domain this metric does not necessarily match the distance in the environment space. Therefore, we look for alternatives that suit our requirements: we derive a different reward from the mutual information, and we study alternatives to the variational approach.

This work focuses on learning meaningful representations, discovering skills and training latent-conditioned policies. In any of the cases, our methodology does not require any supervision and works directly from pixel observations. Additionally, we also study the impact of extra input information in the form of position coordinates. Our proposal is tested over the MineRL (Guss et al., 2021) environment, which is based on the popular Minecraft videogame. Even though the game proposes a final goal, Minecraft is well known by the freedom that it gives to the players, and actually most human players use this freedom to explore this virtual world following their intrinsic motivations. Similarly, we aim at discovering skills in Minecraft without any extrinsic reward.

We generate random trajectories in Minecraft maps with little exploratory challenges, and also study contrastive alternatives that exploit the temporal information throughout a trajectory. The contrastive approach aims at learning an embedding space where observations that are close in time are also close in the embedding space. A similar result can be achieved by leveraging the agents' relative position in the form of coordinates. In the latter, the objective is to infer skills that do not fully rely on pixel resemblance, but also take into account temporal and spatial relationships.

Our final goal is to discover and learn skills that can be potentially used in more broad and complex tasks. Either by transferring the policy knowledge or by using hierarchical approaches. Some works have already assessed this idea specially in robotics (Florensa et al., 2017) or 2D games (Campos et al., 2021). Once the pre-training stage is completed and the agent has learned some basic behaviours or skills, the agent is exposed to an extrinsic reward. These works show how the agents leverage the skill knowledge to learn much faster and encourage proper exploration of the environment in unrelated downstream tasks. However, transferring the policy knowledge is not as straightforward as in other deep learning tasks. If one wants to transfer behaviours (policies), the change in the task might lead to catastrophic forgetting.

Our contributions are the following:

- We demonstrate that variational techniques are not the only ones capable of maximizing the mutual information between inputs and latent variables by leveraging contrastive techniques.
- We provide alternatives for discovering and learning skills in procedurally generated maps by leveraging the agents coordinate information.
- We succesfully implement the reverse form of the mutual information for optimizing pixel-based agents in a complex 3D environment.

2. Related Work

Intrinsic Motivations (IM) are very helpful mechanisms to deal with sparse rewards. In some environments the extrinsic rewards are very difficult to obtain and, therefore, the agent does not receive any feedback to progress. In order to drive the learning process without supervision, we can derive intrinsic motivations as proxy rewards that guide the agents towards the extrinsic reward or just towards better exploration.

Skill Discovery. We relate Intrinsic Motivations to the concept of *empowerment* (Salge et al., 2014), a RL paradigm in which the agent looks for the states where it has more control over the environment. Mohamed and Rezende (2015) derived a variational lower bound on the mutual information that allows to maximize empowerment. Skill discovery extends this idea from the action level to temporally-extended actions. Florensa et al. (2017) merges skill discovery and hierarchical architectures. They learn a high-level policy on top of some basic skills learned in a task-agnostic way. They show how this set-up improves the exploration and enables faster training in downstream tasks. Similarly, Achiam et al. (2018) emphasize in learning the skills dynamically using a curriculum learning approach, allowing the method to learn up to a hundred of skills. Instead of maximizing the mutual information between states and skills they use skills and whole trajectories. Eysenbach et al. (2018) demonstrates that learned skills can serve as an effective pretraining mechanism for robotics. Our work follows their approach regarding the use of a categorical and uniform prior over the latent variables. Campos et al. (2020) exposes the lack of coverage of previous works. They propose *Explore, Discover and Learn* (EDL), a method for skill discovery that breaks the dependency on the distributions induced by the policy. Warde-Farley et al. (2018) provides an algorithm for learning goal-conditioned policies using an imitator and a teacher. They demonstrate the effectiveness

of their approach in pixel-based environments like Atari, DeepMind Control Suite and DeepMind Lab.

Intrinsic curiosity. In a more broad spectrum we find methods that leverage intrinsic rewards that encourage exploratory behaviours. Pathak et al. (2017) present an *Intrinsic Curiosity Module* that defines the curiosity or reward as the error predicting the consequence of its own actions in a visual feature space. Similarly, Burda et al. (2018) uses a Siamese network where one of the encoder tries to predict the output of the other. The bonus reward is computed as the error between the prediction and the random one.

Goal-oriented RL. Many of the works dealing with skill-discovery end up parameterizing a policy. This policy is usually conditioned in some goal or latent variable $z \sim Z$. The goal-conditioned policy formulation along with function approximation was introduced by Schaul et al. *Hindsight Experience Replay* (HER) by Andrychowicz et al. (2017) allows sample-efficient learning from environments with sparse rewards. HER assumes that any state can be a goal state. Therefore, leveraging this idea, the agent learns from failed trajectories as if the final state achieved was the goal state. Trott et al. (2019) proposes to use pairs of trajectories that encourage progress towards a goal while learns to avoid local optima. Among different environments, *Sibling Rivalry* is evaluated in a 3D construction task in Minecraft. The goal of the agent is to build a structure by placing and removing blocks. They use the number of block-wise differences as a naive shaped reward which causes the agent to avoid placing blocks. Simply by adding Sibling Rivalry they manage to improve the degree of construction accuracy.

Representation Learning in RL. In RL we seek for low-dimensional state representations that preserve all the information and variability of the state space in order to make decisions that eventually maximize the reward. This becomes crucial when dealing with pixel-based environments. Ghosh et al. (2018) aims to capture those factors of variation that are important for decision making and are aware of the dynamics of the environment without the need for explicit reconstruction. Lee et al. (2020) also makes use of variational inference techniques for estimating the posterior distribution, but breaks the Markovian assumption by conditioning the probability of the latent variables with past observations. Oord et al. (2018) leverages powerful autoregressive models and negative sampling to learn a latent space from high-dimensional data. This work introduces the InfoNCE loss based on Noise Contrastive Estimation. The intuition behind is that we learn a latent space that allows to classify correctly positive pairs while discriminating from negative samples. We make use of this loss in our contrastive experiments, as well as the following works explained. Laskin et al. (2020) trains an end-to-end model by performing off-policy learning on top of extracted features. This features are computed using a contrastive approach based on pairs of augmented observations, while Stooke et al. (2020) picks pairs of delayed observations. All these works learn representations from pixel-based observations, except for the first one by Ghosh et al. (2018).

3. Information-theoretic skill discovery

Intrinsic Motivations (IM) can drive the agents learning process in the absence of extrinsic rewards. With IM, the agents do not receive any feedback from the environment, but must autonomously learn the possibilities available in it. To achieve that, the agents aim to gain resources and influence on what can be done in the environment. In the *empowerment* (Salge et al., 2014) framework, an agent will look for the state in which has the most control over the environment. This concept usually deals with simple actions. In contrast, *skill-discovery* temporally abstracts these simple actions to create high-level actions that are dubbed as *skills* or *options*. Skill-discovery is formulated as maximizing the mutual information between inputs and skills. This encourages the agent to learn skills that derive as many different input sequences as possible, while avoiding overlap between sequences guided by different skills. Therefore, *skill-discovery* can ease the exploration in complex downstream tasks. Instead of executing random actions, the agent can take advantage of the learned behaviours in order to perform smarter moves towards states with potential extrinsic rewards.

In the next section we formulate the mathematical framework that is used through our work. First we define the classic Markov decision process that typically provide a mathematical formulation for reinforcement learning tasks. Later on, we introduce the tools from information-theory that allows us to define the maximization of the mutual information.

3.1. Preliminaries

Let us consider a Markov decision process (MDP) without rewards as $\mathcal{M} = (\mathcal{S}, \mathcal{A}, \mathcal{P})$. Where $\mathcal{S}$ is the high-dimensional state space (pixel images), $\mathcal{A}$ refers to the set of actions available in the environment and $\mathcal{P}$ defines the transition probability $p(s_{t+1}|s_t, a)$. We learn latent-conditioned policies $\pi(a|s, z)$, where the latent $z \in \mathcal{Z}$ is a random variable.

Given the property of symmetry, the mutual information ($\mathcal{I}$) can be defined using the Shannon Entropy ($\mathcal{H}$) in two ways (Gregor et al. (2017)):

$$\begin{aligned} \mathcal{I}(S, Z) &= \mathcal{H}(Z) - \mathcal{H}(Z|S) \quad \rightarrow \quad \text{reverse} \\ &= \mathcal{H}(S) - \mathcal{H}(S|Z) \quad \rightarrow \quad \text{forward} \end{aligned} \tag{1}$$

For instance, we derive the reverse form of the mutual information (MI):

$$\mathcal{I}(S, Z) = E_{s,z\sim p(z,s)}[\log p(z|s)] - E_{z\sim p(z)}[\log p(z)] \tag{2}$$

The posterior $p(z|s)$ is unknown due to the complexity to marginalize the evidence $p(s) = \int p(s|z)p(z)dz$. Hence, we approximate it with $q_\phi(z|s)$ by using variational inference or contrastive techniques. As a result, we aim at maximizing a lower bound based on the KL divergence. For a detailed derivation we refer the reader to the original work from Mohamed and Rezende (2015).

Therefore, the MI lower bound becomes:

$$\mathcal{I}(S, Z) \geq E_{s,z\sim p(z,s)}[\log q_\phi(z|s)] - E_{z\sim p(z)}[\log p(z)] \tag{3}$$

The prior $p(z)$, can either be learned through the optimization process (Gregor et al., 2017), or fixed uniformly beforehand (Eysenbach et al., 2018). In our case, since we want to maximize the uncertainty over $p(z)$, we define a categorical uniform distribution.

3.1.1. Variational Inference

Variational inference is a well-known method for modelling posterior distributions. Its main advantage is that it avoids computing the marginal $p(s)$, which is usually impossible. Instead, one selects some tractable families of distributions q as an approximation of p.

$$p(z|s) \approx q(z|\theta) \tag{4}$$

We fit q with sample data to learn the distribution parameters θ. In particular, we make use of Variational Auto-Encoders (VAE) (Kingma and Welling (2014)) with categorical classes, namely VQ-VAE (van den Oord et al., 2017), for modelling the mappings from inputs to latents and backwards. The encoder $q_\phi(z|s)$ models $p(z|s)$ and the decoder $q_\psi(s|z)$ models $p(s|z)$.

3.1.2. Contrastive Learning

Contrastive learning is a subtype of self-supervised learning that consists in learning representations from the data by comparison. This field has quickly evolved in the recent years, especially in computer vision. Within this context, the comparisons are between pairs of images. Examples of positive pairs may be augmented versions of a given image like crops, rotations, color transformations, etc; and the negative pairs maye be the other images from the dataset. In this case, representations are learned by training deep neural networks to distinguish between positive or negative pairs of observations. The main diference between contrastive self-supervised (or unsupervised) learning and metric learning is that the former does not require any human annotation, while the later does. Learning representations with no need of labeling data allows scaling up the process and overcoming the main bottleneck when training deep neural networks: the scarcity of supervisory signals.

Similar to what is done in variational techniques, we can compute the mutual information between inputs and latents. In this case, instead of learning the latents by the reconstruction error, they are learned by modeling global features between positive and negative pairs of images. Intuitively, two distinct augmentations (positive pair) should be closer in the embedding space than two distinct images (negative pair) from the dataset. Therefore, we need two encoders in paralel instead of an encoder-decoder architecture. The second encoder is usually called momentum encoder (Chen et al. (2021)) and its weights are updated using exponential moving averages of the main encoder.

In each training step, we forward a batch of different original images through the main encoder while we forward the positive pairs of each of those images through the momentum encoder. In a batch of N samples, we have for each positive pair, $N-1$ negative pairs. We define z as the output of the encoder and z' as the output of the momentum encoder. Then, e is a convolutional neural encoder and h is a projection head (small multi-layer perceptron) that returns a latent z, $(z = h(e_\theta(s)))$.

At the output of the network we can perform a categorical cross-entropy loss where the correct classes are the positive pairs in the batch. The correct classes are in the diagonal of the resulting matrix $z^T W z'$, where W is a projection matrix learned during training. In the other positions, we have the similarity between negative pairs in the embedding space. Minimizing this loss, whose name is InfoNCE (Oord et al., 2018), will encourage the model to find global features that can be found in augmented versions from an image. Moreover, as stated by their authors Oord et al. (2018), minimizing the InfoNCE Eq. 5 loss maximizes the mutual information between inputs and latents. As a result, we learn the desired mapping from input states s to latent vectors z.

$$\mathcal{L}_{InfoNCE} = \log \frac{\exp(z^T W z')}{\exp(z^T W z') + \sum_{i=0}^{K-1} \exp(z^T W z'_i)} \tag{5}$$

4. Methodology

In this section we show the implementation details adopted for each of the stages of our method.

4.1. Exploration

EDL (Explore, Discover and Learn) (Campos et al., 2020) provides empirical and theoretical analysis of the lack of coverage of some methods leveraging the mutual information for discovering skills. Either by using the forward or the reverse form of the MI, the reward given to novel states is always smaller than the one given to known states. The main problem dwells in the induced state distributions used to maximize the mutual information. Most works reinforce already discovered behaviours since they induce the state distribution from a random policy, $p(s) \approx \rho_\pi(s) = E_z[\rho_\pi(s|z)]$. EDL is agnostic to how $p(s)$ is obtained, so we can infer it in very different ways. One manner would be to induce it by leveraging a dataset of expert trajectories that will encode human-priors that are usually learned poorly with information-theoretic objectives. Another way is to make use of an exploratory policy that is able to induce a uniform distribution over the state space. Since this is not clearly solved in complex 3D environments yet, we explore by using random policies in bounded maps.

4.2. Skill-Discovery

In Section 3, we proposed two distinct approaches for modelling the mapping from the observations s to the latent variables z: variational inference and contrastive learning. The skill-discovery pipelines for both approaches are depicted in Figure 1, and are described in the remaining of this section.

4.2.1. Variational Inference

Vector-Quantized VAE (VQ-VAE) (van den Oord et al., 2017) is a variational model that allows us to have a categorical distribution $p(z)$ as a bottleneck (also called codebook), an encoder q_ψ that estimates the posterior $p(z|s)$, and the decoder q_ϕ that estimates $p(s|z)$.

Before training, the model requires to fix the length of the codebook. This determines the granularity of the latent variables. If we choose a large number of codes, we will end up with latent variables that encode very similar states. Instead, if we choose a small number, we will encourage the model to find latents that generalize across diverse scenarios. The perplexity metric measures the number of codewords needed to encode our whole state distribution. In practice, this metric is computed per batch during training. Since it is not possible to know beforehand the number of useful latent variables that our model will discover, one can iterate over different codebook lengths until they find a good trade-off between generalization and granularity.

In EDL (Campos et al., 2020), the purpose of training a variational auto-encoder was to discover the latent variables that condition the policies in the Learning stage. But the learned representations were discarded. Instead, in our case, we also leverage the representations learned in the encoder and decoder for training the RL agent. This allows for faster and more efficient trainings of the RL agents.

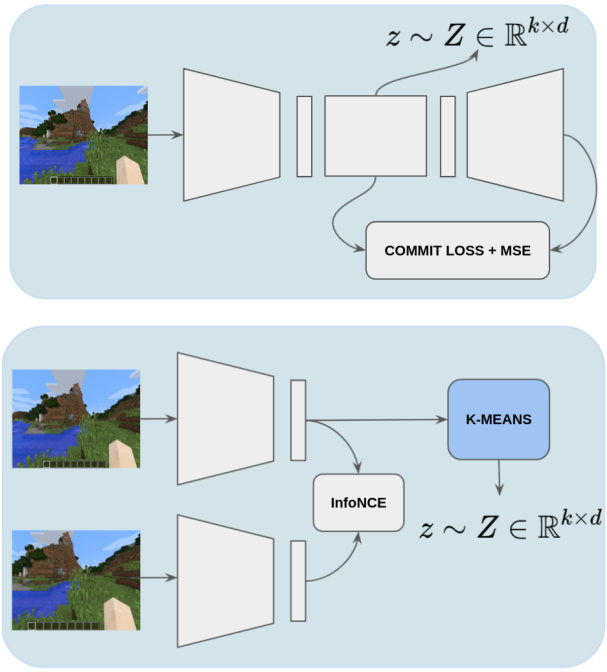


Figure 1. Skill-discovery pipeline with (top) variational and (bottom) contrastive approaches.

4.2.2. Contrastive Learning

In the contrastive case, we follow the idea of Stooke et al. (2020) where the positive pairs are delayed observations from a specific trajectory. Once the latents are learned, we define a categorical distribution over latents by clustering the embedding space of the image representations (Yarats et al., 2021). This step is performed using K-Means with K clusters equal to the length of the VQ-VAE codebook.

At this point, we have the same set-up for both the contrastive and variational approaches. The K-Means centroids are equivalent to the VQ-VAE codebook embeddings, so we can maximize the mutual information between the categorical latents and the inputs with the same Equation 2.

4.3. Skill-Learning

In the last stage of the process, we aim to train a policy $\pi(a|s, z)$ that maximizes the mutual information between inputs and the discovered latent variables. At each episode of the training, we sample a latent variable $z \sim p(z)$. Then, at each step, this embedding is concatenated with the embedding of the encoded observation at timestep t and forwarded

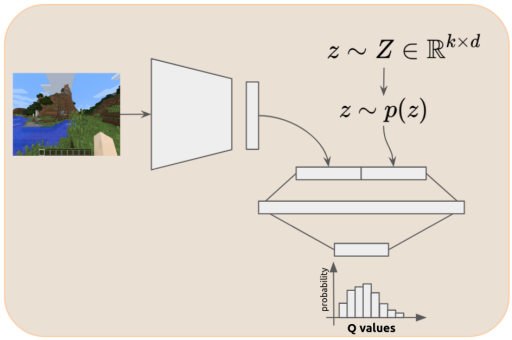


Figure 2. Skill-Learning pipeline for both contrastive and variational approaches.

through the network. This process is depicted in Figure 2. At this stage, the latent variables are interpreted as navigation goals. Hence, the agent is encouraged to visit those states that brings it closer to the navigation goal or latent variable. Since each of the latent variables encode different parts of the state distribution, this results in covering different regions for each z. Our method adopts the reverse form of the mutual information. Since we fix $p(z)$ as a uniform distribution we can remove the $\log p(z)$ constant term from the Equation 3, which results in the reward of Equation 7 for the variational case, and Equation 8 for the contrastive case. The agent receives a reward of 1 only if the embedding that is conditioning the policy is the closest to the encoded observation.

$$r(s, z) = q_\phi(z|s) \tag{6}$$

$$q_\phi(z = k|s) = \begin{cases} 1, & \text{if} \quad k = argmin_j ||z_e(s) - e_j|| \\ 0, & \text{otherwise} \end{cases} \tag{7}$$

Where $z_e(s)$ is the encoded observation and e is the codebook of embeddings.

$$q_\phi(z = k|s) = \begin{cases} 1, & \text{for } k = argmax_j (z_e(s) W e_j) \\ 0, & \text{otherwise} \end{cases} \tag{8}$$

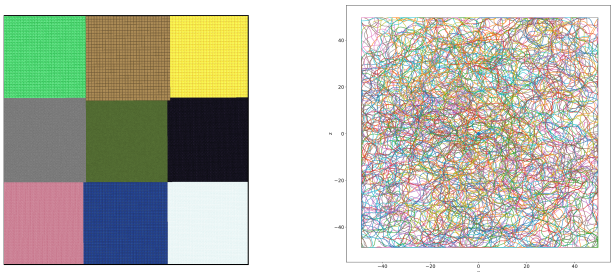
In the contrastive case, we compute a similarity measure. Therefore, we look for the latent index with highest similarity to the current observation.

5. Experiments

In this section we asses our method in two different Minecraft maps and set-ups. In each of the trainings we follow the pipeline described in Section 4, where we first generate random trajectories in the specified map. Later, we discover some categorical latents from them, and finally we train some conditioned policies on these latents. We refer the reader to the Appendix A to check the hyperparameters used in the different methods.

5.1. Discovery and learning of skills in handcrafted maps

The first map, in Figure 3a, is a 2D handcrafted map that consists in 9 different regions where the only difference among them is the floor type. The $p(s)$ induced from the random trajectories seen in Figure 3b covers the state subspace uniformly.



(a) (b)

Figure 3. **(a)** Top-view of the toy map used for asses our method. The dimensions of this map are 100 square meters, if we consider that the Minecraft coordinate system was measured in meters. The agent has no exploratory issues even with a random policy. **(b)** Trajectories generated by a random agent deployed in any region of the map.

The *index maps* shown in Figure 4 are generated from the random trajectories in Figure 3b. We encode each observation from each trajectory with the index of the closest embedding in the latent codebook or K-Means centroids. We assign a color to each of the indexes and we get what we refer as an *index map*. Thanks to these maps, it is very easy to visualize whether the learned embeddings are meaningful or not.

Figure 4 compares the index maps learned with the varia-

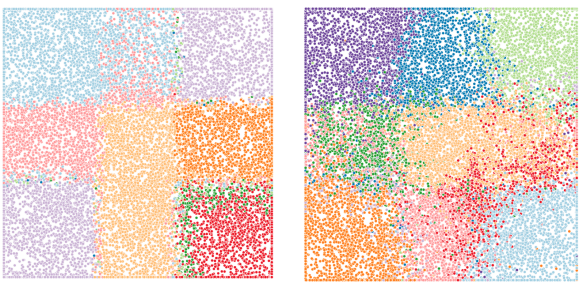
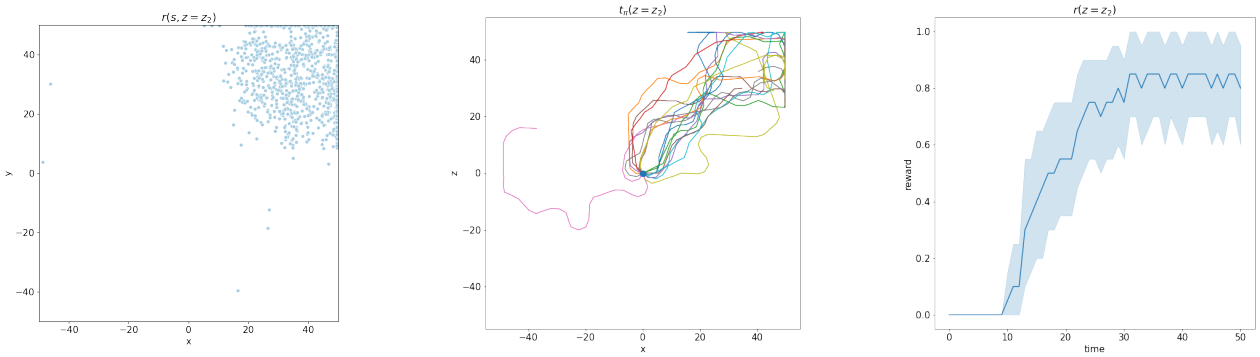


Figure 4. **Left:** Index map generated by the variational approach. **Right:** Index map generated by the contrastive approach.



(a) (b) (c)

Figure 5. **(a)** Observations that are encoded as the latent vector 2. **(b)** shows the trajectories performed by the agent and **(c)** the average reward received in those trajectories. Both plots conditioning the agent on the latent vector 2.

tional and constrastive methods. In general we can see that the variational model discretizes slightly better, however it is unable to distinguish between two particular regions. Instead, despite the mixing and overlapping among different discovered latents, the contrastive approach manages to clusterize these regions decently. When studying these visualizations we should be aware that we are not taking into account the direction in which the agent was looking at the time it was recorded. Therefore, two points that are next to each other, might be encoding totally different views and hence will result in different latent codes.

Learning representations from these observations is quite straightforward, since the dynamics are not very complex. Despite that, either when training with the contrastive and variational methods, we usually find latent centroids that encode observations in between plots. For this reason, if we wanted to have a categorical latent vector for each of the regions we could overclusterize the latent embeddings (in the contrastive case) or select a larger codebook (in the variational case). Since we train the agent in a map where we have already collected observations from random policies, we can plot the reward that those observations would be given when conditioned in each of the categorical latents. For instance, in Figure 5a we can see that the observations recorded in the top-right corner are encoded as the second discovered latent. Therefore, if we condition the policy with it, the agent will receive a reward of 1, following Equation 6.

In Figure 5b, we show how the performance of the agent when conditioned in this skill. All the trajectories head to the navigation goal except for one. Moreover, regardless of the initial direction, we can see that the agent is capable of orientating and go to the right direction. Lastly, in Figure 5c, we can see how it requires 10 steps to start triggering the intrinsic reward until reaching a stable 0.8 reward in average. For more examples on other latent codes we refer the reader to the Appendix B.

5.2. Discovery and learning of skills in realistic maps

The approach presented in the simple and handcrafted map of Section 5.1 fails when tested in more realistic Minecraft maps, like the one in Figure 6a. In this case, the discovered skills depicted in Figure 6b are no longer distributed in separable clusters over the map, and one skill (in light blue) tends to dominate over the others. Since we are treating each latent variable as a navigation goal, the ones that are spread out over the whole map will not guide but confuse the agent as it is receiving high reward in multiple places.

We overcome this problem by exploiting the spatial information provided by the environment in the form of coordinates. The learned embedding space contains a relationship between visual features and relative spatial coordinates with respect to the initial state, always the same one. This allows our agent to distinguish between two visually identical mountains located at two different positions in the map. Figure 6 shows complementary skills discovered from pixels (Figure 6b) or coordinates (Figure 6c). We would like to discover skills that take into account not only the visual similarity but also the position relative to the initial state, so we adopt a solution that considers the two types of state representations (Figure 6d).

We scale the VQ-VAE (van den Oord et al., 2017) to encode both pixels and coordinates independently. We sum up their embeddings and quantize the result to one of the codebook embeddings e. Now, the decoder of the model contains two independent branches, where one reconstructs an image and the other reconstructs a coordinate from the same embedding. The only difference from the original loss function is that we add a term for learning the coordinate reconstruction.

Once we have better latent embeddings for realistic maps,

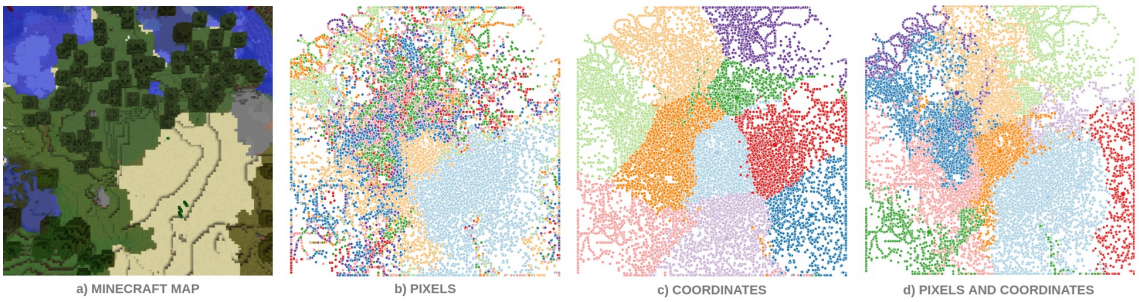


Figure 6. a) Top-view of our Minecraft map. The caption below b) c) d) refers to the nature of the states (pixels, coordinates or both) where all the trajectories were collected by random policies. Each coloured point indicates the closest centroid to the encoded embedding.

we can train an agent conditioned on them. Qualitative results are presented in Figure 7, where we can see an example of a latent encoding the bottom-left observations and the trajectories and rewards received in 10 evaluation episodes. The performance of all the conditioned policies is included in the Appendix C. We observed that at least 6 skills out of 10 could be potentially leveraged in a hierarchical policy.

Figure 8 shows a clear example of the pixels and coordinates mixing. If we based our encoder in just pixel observations we would have an almost identical latent encoding for both images. Instead, thanks to the coordinates, they become two separate latents. This may help the agent in being more precise and distinguish different regions of the space despite its visual similarity.

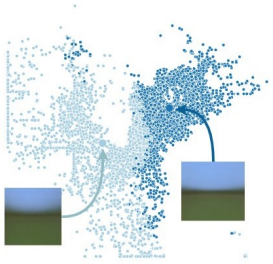
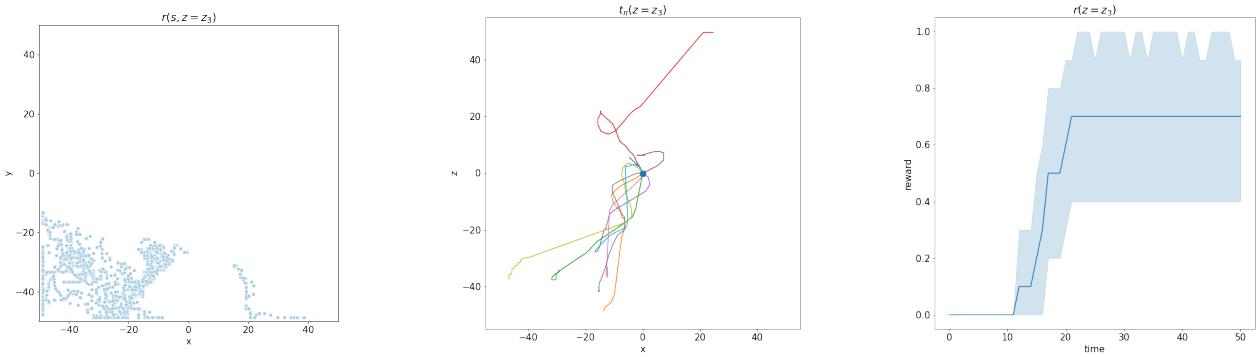


Figure 8. A very homogeneous and big region of the space becomes two different skills when using a combination of pixels and coordinates.

6. Conclusions

In this work we present a framework to extend the EDL paradigm to pixels. This requirement presents one main problem. Following the EDL derivation of the mutual information, we optimize the agents based on the MSE, which does not have any notion of space in the environment. To overcome that, we propose to derive the reverse form of the mutual information between inputs and latents. In this manner we compute the intrinsic reward in the embedding space which encodes the dynamics learned during the skill-discovery phase. We observed that contrastive and variational approaches are capable of learning the mapping from inputs to latents. In the former case, we leverage the use of positive pairs based on delayed observations, and the K-



a) b) c)

Figure 7. a) Observations that are encoded as the latent vector 3. b) and c) show the trajectories performed by the agent and the reward received in average when conditioned on skill 3.

Means clustering algorithm for determining the categorical latents. In the latter, we make use of a VQ-VAE model that directly allows to infer categorical latents from the input data. We qualitatively show that some skills are correctly learned, specially in the toy maps where the environment dynamics are much easier. We demonstrate that for learning useful navigation goals in realistic maps we need more information, and therefore, we learn an embedding space combining pixel and coordinates observations.

References

J. Achiam, H. Edwards, D. Amodei, and P. Abbeel. Variational option discovery algorithms, 2018.

I. Akkaya, M. Andrychowicz, M. Chociej, M. Litwin, B. McGrew, A. Petron, A. Paino, M. Plappert, G. Powell, R. Ribas, et al. Solving rubik's cube with a robot hand. *arXiv preprint arXiv:1910.07113*, 2019.

M. Andrychowicz, F. Wolski, A. Ray, J. Schneider, R. Fong, P. Welinder, B. McGrew, J. Tobin, P. Abbeel, and W. Zaremba. Hindsight experience replay. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pages 5055–5065, 2017.

C. Berner, G. Brockman, B. Chan, V. Cheung, P. Debiak, C. Dennison, D. Farhi, Q. Fischer, S. Hashme, C. Hesse, et al. Dota 2 with large scale deep reinforcement learning. *arXiv preprint arXiv:1912.06680*, 2019.

T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, et al. Language models are few-shot learners. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, 2020.

Y. Burda, H. Edwards, A. Storkey, and O. Klimov. Exploration by random network distillation. In *International Conference on Learning Representations*, 2018.

V. Campos, A. Trott, C. Xiong, R. Socher, X. Giro-i Nieto, and J. Torres. Explore, discover and learn: Unsupervised discovery of state-covering skills. In *International Conference on Machine Learning*, pages 1317–1327. PMLR, 2020.

V. Campos, P. Sprechmann, S. Hansen, A. Barreto, S. Kapturowski, A. Vitvitskyi, A. P. Badia, and C. Blundell. Coverage as a principle for discovering transferable behavior in reinforcement learning. *arXiv preprint arXiv:2102.13515*, 2021.

M. Caron, H. Touvron, I. Misra, H. Jégou, J. Mairal, P. Bojanowski, and A. Joulin. Emerging properties in self-supervised vision transformers. *arXiv preprint arXiv:2104.14294*, 2021.

X. Chen, S. Xie, and K. He. An empirical study of training self-supervised visual transformers. *arXiv preprint arXiv:2104.02057*, 2021.

K. Cobbe, C. Hesse, J. Hilton, and J. Schulman. Leveraging procedural generation to benchmark reinforcement learning. In *International conference on machine learning*, pages 2048–2056. PMLR, 2020.

A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021.

B. Eysenbach, A. Gupta, J. Ibarz, and S. Levine. Diversity is all you need: Learning skills without a reward function. In *International Conference on Learning Representations*, 2018.

C. Florensa, Y. Duan, and P. Abbeel. Stochastic neural networks for hierarchical reinforcement learning. In *International Conference on Learning Representations*, 2017.

D. Ghosh, A. Gupta, and S. Levine. Learning actionable representations with goal conditioned policies. In *International Conference on Learning Representations*, 2018.

K. Gregor, D. J. Rezende, and D. Wierstra. Variational intrinsic control. In *International Conference on Learning Representations (workshop)*, 2017.

W. H. Guss, M. Y. Castro, S. Devlin, B. Houghton, N. S. Kuno, C. Loomis, S. Milani, S. Mohanty, K. Nakata, R. Salakhutdinov, et al. The minerl 2020 competition on sample efficient reinforcement learning using human priors. *arXiv preprint arXiv:2101.11071*, 2021.

J. Hwangbo, I. Sa, R. Siegwart, and M. Hutter. Control of a quadrotor with reinforcement learning. *IEEE Robotics and Automation Letters*, 2(4):2096–2103, 2017.

D. P. Kingma and M. Welling. Auto-encoding variational bayes, 2014.

M. Laskin, A. Srinivas, and P. Abbeel. Curl: Contrastive unsupervised representations for reinforcement learning. In *International Conference on Machine Learning*, pages 5639–5650. PMLR, 2020.

A. Lee, A. Nagabandi, P. Abbeel, and S. Levine. Stochastic latent actor-critic: Deep reinforcement learning with a latent variable model. *Advances in Neural Information Processing Systems*, 33, 2020.

S. Mohamed and D. J. Rezende. Variational information maximisation for intrinsically motivated reinforcement learning. In *Proceedings of the 28th International Conference on Neural Information Processing Systems-Volume 2*, pages 2125–2133, 2015.

A. v. d. Oord, Y. Li, and O. Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.

D. Pathak, P. Agrawal, A. A. Efros, and T. Darrell. Curiosity-driven exploration by self-supervised prediction. In *International Conference on Machine Learning*, pages 2778–2787. PMLR, 2017.

C. Salge, C. Glackin, and D. Polani. Empowerment–an introduction. In *Guided Self-Organization: Inception*, pages 67–114. Springer, 2014.

T. Schaul, D. Horgan, K. Gregor, and D. Silver. Universal value function approximators.

J. Schrittwieser, I. Antonoglou, T. Hubert, K. Simonyan, L. Sifre, S. Schmitt, A. Guez, E. Lockhart, D. Hassabis, T. Graepel, et al. Mastering atari, go, chess and shogi by planning with a learned model. *Nature*, 588(7839): 604–609, 2020.

A. Stooke, K. Lee, P. Abbeel, and M. Laskin. Decoupling representation learning from reinforcement learning. *arXiv preprint arXiv:2009.08319*, 2020.

R. S. Sutton and A. G. Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

A. Trott, S. Zheng, C. Xiong, and R. Socher. Keeping your distance: Solving sparse reward tasks using self-balancing shaped rewards. In *Proceedings of the International Conference on Neural Information Processing Systems*, 2019.

A. van den Oord, O. Vinyals, and K. Kavukcuoglu. Neural discrete representation learning. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pages 6309–6318, 2017.

O. Vinyals, I. Babuschkin, W. M. Czarnecki, M. Mathieu, A. Dudzik, J. Chung, D. H. Choi, R. Powell, T. Ewalds, P. Georgiev, et al. Grandmaster level in starcraft ii using multi-agent reinforcement learning. *Nature*, 575(7782): 350–354, 2019.

D. Warde-Farley, T. Van de Wiele, T. Kulkarni, C. Ionescu, S. Hansen, and V. Mnih. Unsupervised control through non-parametric discriminative rewards. *arXiv preprint arXiv:1811.11359*, 2018.

D. Yarats, R. Fergus, A. Lazaric, and L. Pinto. Reinforcement learning with prototypical representations. *arXiv preprint arXiv:2102.11271*, 2021.

A. Hyperparameters

VQ-VAE	
learning rate	1e-3
batch size	256
num hiddens	64
num residual hiddens	32
num residual layers	2
embedding dim	256
num embeddings	10
β	0.25
decay	0.99

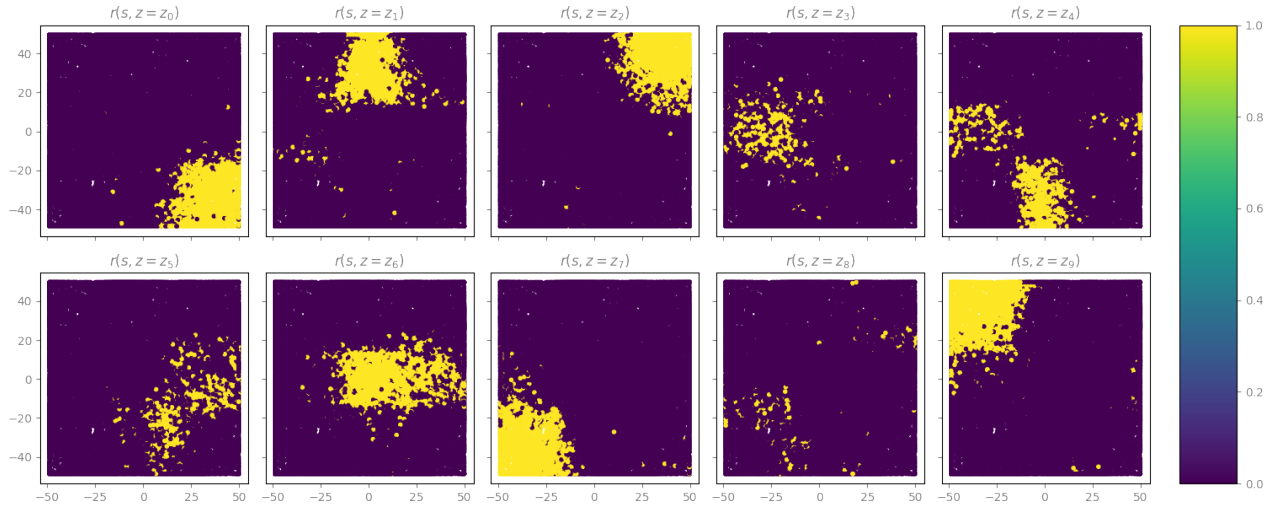
Contrastive	
learning rate	1e-3
batch size	128
τ	5e-3
soft update	2
embedding dim	128
k_μ	15
k_σ	5

Rainbow	
learning rate	2.5e-4
batch size	64
sampling	uniform
max episode steps	500
update interval	4
frame skip	10
gamma	0.99
clip delta	yes
num step return	1
adam eps	1.e-8
gray scale	no
frame stack	no
final expl frames	3.5e5
final epsilon	0.01
eval epsilon	0.001
replay capacity	10.e6
replay start size	1.e4
prioritized	yes
target upd interval	2.e4

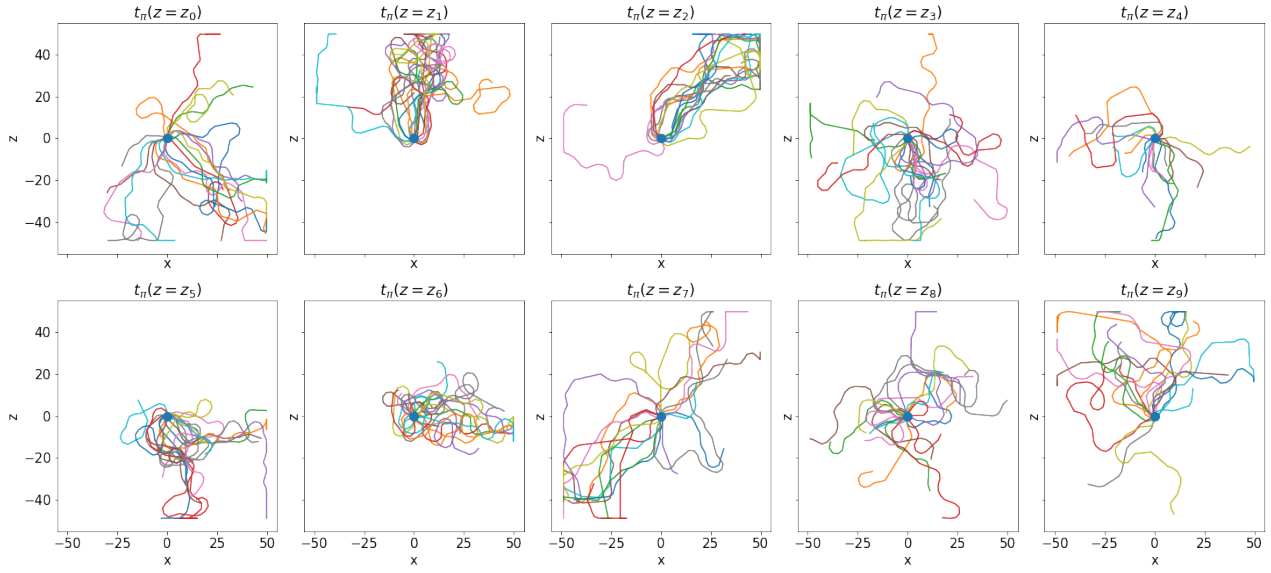
B. Learning from Pixels with Contrastive approach

(Handcrafted map)

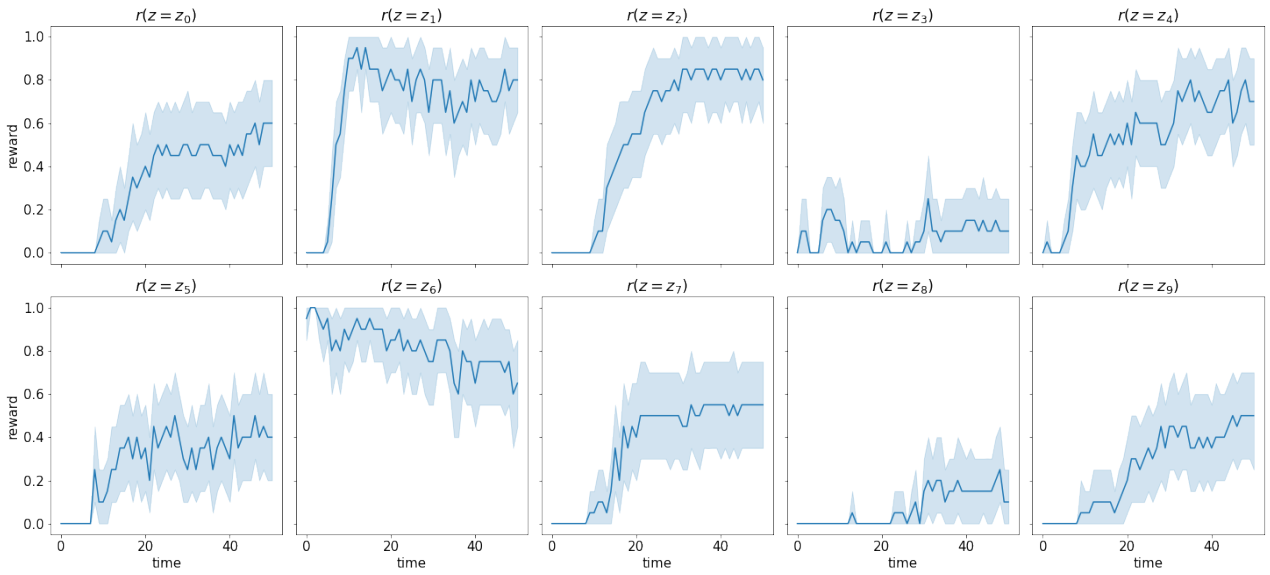
In this first plot, we show which observations would give a positive reward for each conditioning latent. In some cases they encode a very specific region, however, in some others the reward is spread over the map leading to poor policies.



As we can see in the evaluated trajectories, latents 0,1,2,4,5,6 are properly learned. Instead, we can see that latents 3,7,8,9 are more ambiguous and the agent does not solve the self-supervised task correctly.



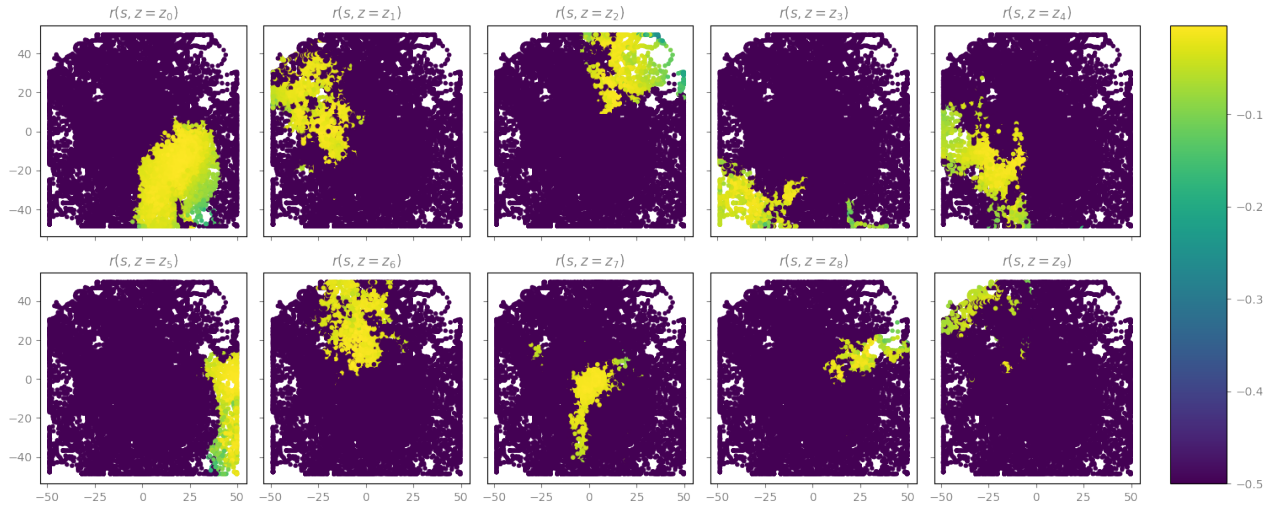
In the last plot we show the average reward of all the trajectories for each of the conditioning latents. We can see that the curves with higher reward match those trajectories in the previous plot that lead to the region encoded by the latent.


$r(z = z_0)$
$r(z = z_1)$
$r(z = z_2)$
$r(z = z_3)$
$r(z = z_4)$
$r(z = z_5)$
$r(z = z_6)$
$r(z = z_7)$
$r(z = z_8)$
$r(z = z_9)$
reward
time
0.0
0.2
0.4
0.6
0.8
1.0
0
20
40

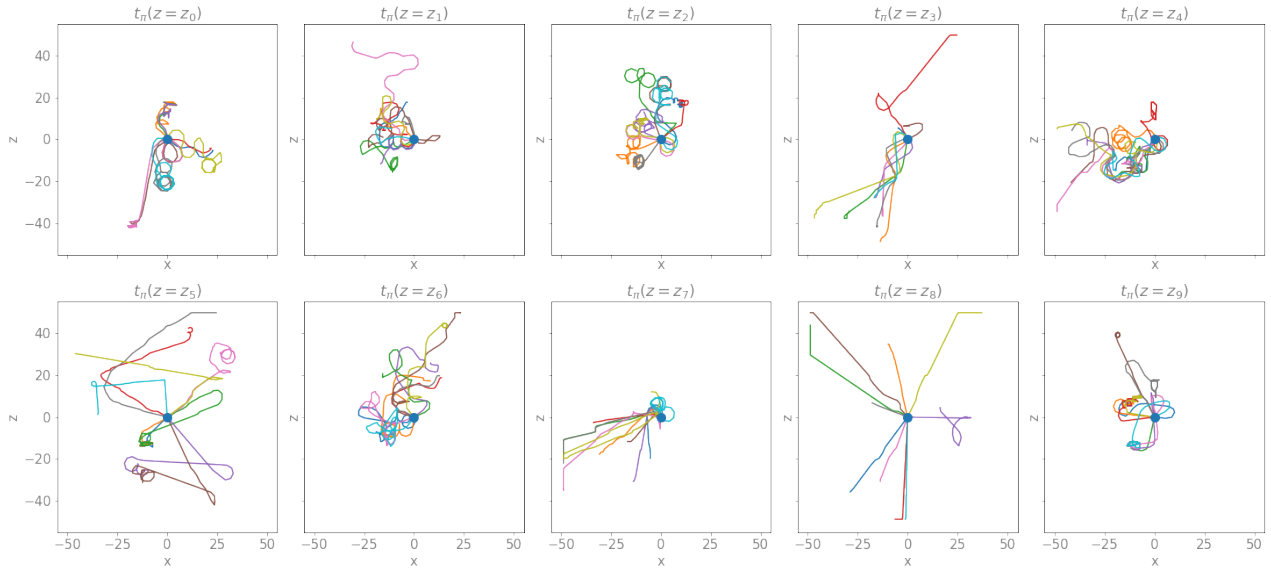
C. Learning from Pixels and Coordinates with Variational approach

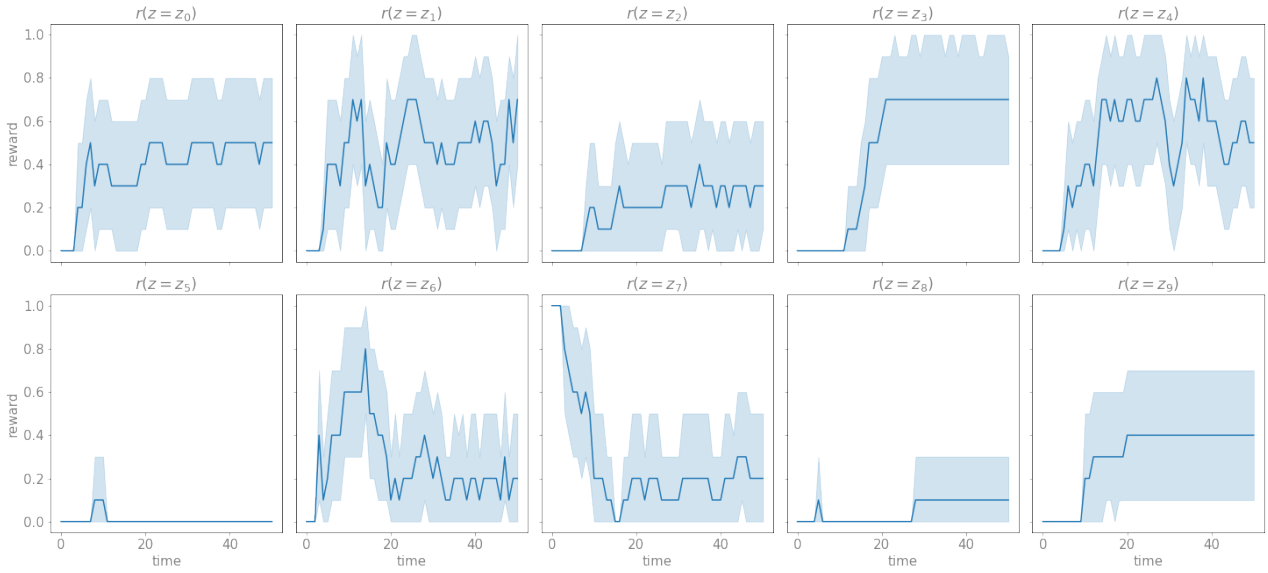
(Realistic map)

In this case, we can see that despite deploying the agent in a realistic map, the discovered latents encode very specific regions of the map. This is achieved by leveraging the raw pixel information and the relative position of the agent respect to the initial point.



Due to the dynamics of this map, we can see that is much more difficult to learn the right policies to guide the agent towards the regions encoded by the latents. Skills 1, 2, 3, 4, 6 and 7 still achieve good results, where, regardless of the agent's initial direction, it manages to reach the desired region.




$r(z = z_0)$
$r(z = z_1)$
$r(z = z_2)$
$r(z = z_3)$
$r(z = z_4)$
$r(z = z_5)$
$r(z = z_6)$
$r(z = z_7)$
$r(z = z_8)$
$r(z = z_9)$
reward
time
0.0
0.2
0.4
0.6
0.8
1.0
0
20
40