Visual Grounding for Object-Level Generalization in Reinforcement Learning

Haobin Jiang[1] and Zongqing Lu[1,2⋆]

[1] School of Computer Science, Peking University
[2] Beijing Academy of Artificial Intelligence
{haobin.jiang,zongqing.lu}@pku.edu.cn

Abstract. Generalization is a pivotal challenge for agents following natural language instructions. To approach this goal, we leverage a vision-language model (VLM) for visual grounding and transfer its vision-language knowledge into reinforcement learning (RL) for object-centric tasks, which makes the agent capable of zero-shot generalization to unseen objects and instructions. By visual grounding, we obtain an object-grounded confidence map for the target object indicated in the instruction. Based on this map, we introduce two routes to transfer VLM knowledge into RL. Firstly, we propose an object-grounded intrinsic reward function derived from the confidence map to more effectively guide the agent towards the target object. Secondly, the confidence map offers a more unified, accessible task representation for the agent's policy, compared to language embeddings. This enables the agent to process unseen objects and instructions through comprehensible visual confidence maps, facilitating zero-shot object-level generalization. Single-task experiments prove that our intrinsic reward significantly improves performance on challenging skill learning. In multi-task experiments, through testing on tasks beyond the training set, we show that the agent, when provided with the confidence map as the task representation, possesses better generalization capabilities than language-based conditioning. The code is available at https://github.com/PKU-RL/COPL.

Keywords: Reinforcement learning · Visual grounding

1 Introduction

In the field of artificial intelligence, the ability of agents to understand and follow natural language instructions in an open-ended manner is crucial [4, 5, 10, 47]. However, the scope of training content for an agent's policy learning is always finite. Zero-shot generalization task which involves being instructed to interact with diverse objects not encountered during training, from the vast realm of human vocabulary, represents a pivotal step towards creating general artificial intelligence systems capable of adapting to a wide range of real-world scenarios [10, 51]. As a popular open-ended 3D game, Minecraft serves as an ideal testbed

⋆ Corresponding author.

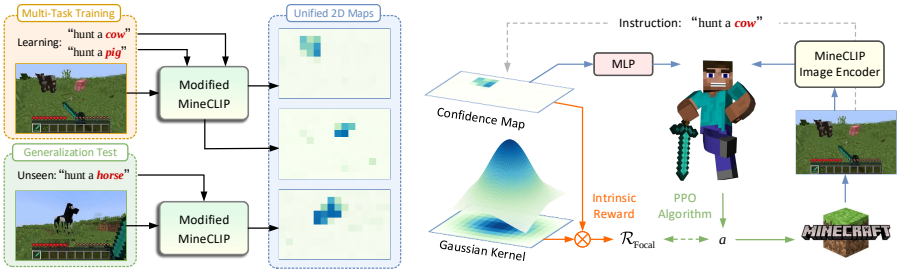


Fig. 1: Overview of **C**LIP-guided **O**bject-grounded **P**olicy **L**earning (**COPL**). (*left*) **Visual grounding**: The instruction is converted into a unified 2D confidence map via our modified MineCLIP. (*right*) **Transfer VLM knowledge into RL**: The agent takes the confidence map as the task representation and is trained with our proposed focal reward derived from the confidence map to guide the agent toward the target.

for learning and evaluating generalization ability. At its core, Minecraft offers procedurally generated worlds with unlimited size and a large variety of tasks ranging from navigation and combat to building and survival [17, 55, 58, 60, 65]. Compared with canonical game environments such as Go [50], Atari [37], and StarCraft [16, 54], Minecraft mirrors the complexity of real-world challenges and offers a wide range of objects and tasks with natural language instructions.

Given the finite data scope for the agent's policy learning, it is infeasible for the policy to directly comprehend the vast array of object names beyond the training set, which human language instructions might contain. To equip an agent with zero-shot generalization ability over objects, integration of a vision-language model (VLM) is a promising way [59]. A VLM aligns images and language vocabularies into the same feature space, bridging the gap between visual observations and natural language instructions. Therefore, it has the capability to ground the agent's unseen text, *e.g.*, names of novel objects, in visual observations, enabling the agent to comprehend instructions not encountered during training. CLIP [42] emerges as a significant model and has become widely used [18, 39, 48, 49]. Recent works have adopted CLIP as a foundation model for open-vocabulary object detection [19, 26, 61] and segmentation [12, 30, 44], leveraging its rich vision-language knowledge. Moreover, CLIP even exhibits remarkable segmentation capabilities and explainability without fine-tuning [29, 64].

CLIP's visual grounding ability to perform segmentation inspires two routes for enhancing the agent's policy learning in Minecraft by transferring VLM knowledge into reinforcement learning (RL). The first approach is *transfer via reward*. The pixel area of the target object can be used as a surrogate for the distance between the agent and the target object. It can then serve as an intrinsic reward [2] to guide the agent towards the target object, thereby facilitating interaction. The second one is *transfer via representation*. The segmentation result can replace the language instruction as a unified, more accessible task representation. Practical research in robotics proves that models with such location input show superior performance compared to mere text input [51]. Most importantly,

the segmentation is open-vocabulary [42, 64], which means it remains effective for instructions containing novel objects not encountered during agent training.

Thanks to MineCLIP [17], a variant of CLIP fine-tuned on Internet-scale Minecraft videos from YouTube, it becomes accessible to develop a generalizable agent in Minecraft, following the aforementioned inspirations. Initially, MineCLIP is merely used as a tool to measure the similarity between a sequence of visual observations and the instruction. It serves as an intrinsic reward for RL and achieves notable performance in Minecraft. Based on this foundation, our goal is to further explore the potential capabilities of MineCLIP, enabling it to offer additional visual grounding information beyond observation-instruction similarity to aid the agent's policy learning and improve its generalization ability.

In this paper, we propose a **C**LIP-guided **O**bject-grounded **P**olicy **L**earning method, namely **COPL**, that transfers the vision-language knowledge contained in MineCLIP to RL at a minimal cost. By visual grounding, we generate a confidence map of the target object indicated in the language instruction via our modified MineCLIP. We extend MineCLIP with modifications inspired by MaskCLIP [64] so that it can segment the specified object from the image. As illustrated in Figure 1 (*left*), our approach can convert instructions into unified two-dimensional confidence maps. To leverage this object-grounded result, we first design an intrinsic reward that takes into account the pixel area and location of the target object in the image observation. By doing so, we also address a deficiency of the original MineCLIP reward [17]: it is insensitive to the distance to the target object [7, 42]. Furthermore, we integrate the resulting confidence map into the policy input as a task representation, as illustrated in Figure 1 (*right*). Based on this, our agent can possess zero-shot generalization capability over objects through multi-task RL trained on *only a limited set of instructions.*

We evaluate COPL on basic skill learning and zero-shot object-level generalization in Minecraft. Firstly, we conduct a group of single-task experiments to show that our object-grounded intrinsic reward successfully enables the agent to acquire various challenging basic skills while the MineCLIP reward fails [7, 17]. Then we extend our evaluation to instruction-following scenarios, where we train the agent with a set of instructions. In our test, the agent exhibits the capacity to execute instructions involving previously unseen targets, effectively demonstrating its generalization ability over objects. COPL's success rate on unseen targets surpasses that of language-conditioned methods by around 300% in the hunting domain and 100% in the harvest domain. Though we implement and evaluate COPL in Minecraft, we believe our method is extendable to other similar open-ended environments and draws insights into transferring VLM knowledge into RL for training generalizable agents.

2 Preliminary

Problem Statement. In this paper, we focus on *object-centric* tasks in Minecraft, where the agent is instructed to interact with diverse objects. By zero-shot *object-level* generalization, we mean that the agent is instructed to interact with objects

beyond the training scope without fine-tuning during evaluation. To formalize, we denote the set of objects with which the agent learns to interact during the training phase as C_t, and the set of objects with which the agent is required to interact during the evaluation phase as C_e. To test the generalization ability of the agent, C_e consists of objects that are not in C_t. For example, during training, the agent learns to accomplish language instructions "`hunt a cow`" and "`hunt a sheep`". However, during evaluation, it will encounter instructions like "`hunt a horse`" or "`hunt a chicken`", where neither `horse` nor `chicken` appears in the instructions during training. Note that we do not consider generalization ability concerning unseen actions, which could be left as future work. Therefore, instructions during evaluation should have the *same behavior patterns* as those learned in training. For instance, when training with "`hunt sth.`" and "`harvest sth.`", testing with "`explore the world`" is not considered.

Zero-Shot Generalization. Our zero-shot object-level generalization in Minecraft is a specific form of broader zero-shot generalization (ZSG) defined in the contextual Markov decision process (CMDP) framework [25] in RL. While ZSG typically involves adapting to new environments or tasks, our focus is on enabling agents to follow instructions and interact with novel objects not encountered during training. This object-level adaptation aligns with ZSG's objective of performing effectively in unseen scenarios. In other words, the "context" in our case is specifically defined as the target objects indicated in instructions.

MineCLIP for Minecraft RL. MineCLIP is a VLM pre-trained on Internet-scale Minecraft videos from YouTube [17], learning the alignment between video clips (16 frames) and natural language. Similar to CLIP [42], MineCLIP adopts a ViT [14] as the image encoder and a GPT [43] as the text encoder. The main difference between MineCLIP and CLIP is that MineCLIP takes as input a sequence of 16 images. Therefore, MineCLIP incorporates an additional module to aggregate the 16 embeddings generated by the image encoder. The proposed two mechanisms include a temporal transformer (MineCLIP[attn]) and direct average pooling (MineCLIP[avg]). In this paper, we choose the former as our base model due to its better performance in Programmatic tasks compared to the latter [17]. For RL in Minecraft, MineCLIP provides an intrinsic reward function $\mathcal{R}_i : \mathcal{G} \times \mathcal{S}^{16} \to \mathbb{R}$, representing the similarity between the image observation sequence of the previous 16 steps $[s_{t-15}, \cdots, s_{t-1}, s_t]$ and the task prompt g.

3 Related Work

Minecraft Research. Broadly, challenges in Minecraft can be categorized into high-level task planning and low-level skill learning. For high-level planning, where agents must make decisions on which skills to employ sequentially based on the given instruction, the field has converged towards leveraging large language models (LLMs) [40, 55, 57, 58, 60, 65]. Regarding learning low-level skills, the difficulty lies in the absence of well-defined dense reward and a vast variety of objects to interact with in Minecraft. Unlike the convergence in high-level planning approaches, two distinct routes have emerged in low-level learning.

The first route, represented by MineCLIP [17], utilizes the reward derived from the alignment between text and video clip or other manually designed rewards for RL [60]. The second one follows the principles of VPT [3], where skills are acquired through imitation learning based on large-scale demonstration [7,8,31]. Our work falls in the scope of low-level skill learning with RL.

Instruction-Following RL. Language has been explored in goal-conditioned RL widely for its compositional structure [33]. This feature allows goal-conditioned policies to better capture the latent structure of the task space and generalize to unseen instructions that combine seen words [9, 11, 21, 36, 41]. With the development of LLM and VLM, language also becomes a means of providing intrinsic rewards in RL. The similarity or correlation between instructions and current states provides dense rewards to guide the agent's learning more effectively [15, 17, 27, 34]. Our work stands out by enabling the policy to generalize to instructions that contain previously unseen targets.

CLIP for Embodied AI. CLIP [42] provides diverse usage for AI research. We categorize these applications into three areas: *encoding*, *retrieving* and *locating*. Encoding, the most common use of CLIP, leverages CLIP encoders to represent images and/or texts [23, 35, 49]. Our work also utilizes the MineCLIP image encoder to process raw image observations. Retrieving mostly involves navigation tasks, where CLIP assists in selecting the most matching image from a set based on the given instruction [6, 10, 13, 47]. The most relevant usage to our work is locating, which applies methods like MaskCLIP [64] or GradCAM [46] on CLIP to determine the position of the specific object in images [18, 56, 63]. Based on the object location, agents can conduct planning with a depth detector [18] or imitation learning [56, 63]. In contrast, our work focuses on training agents via RL with information solely extracted from image observations.

4 Method

In this section, we detail the implementation of our COPL method in Minecraft. We introduce how to exploit the visual grounding capability of MineCLIP through modifications, enabling the segmentation of the target object indicated in the language instruction (Section 4.1). This process yields an object-grounded confidence map, where each element represents the probability of the specified target's presence. Based on this confidence map, we first implement VLM knowledge transfer via reward, presenting a simple but effective object-grounded intrinsic reward to guide the agent toward the target (Section 4.2). Then, we propose transfer via representation, where we integrate the confidence map into the policy as a task representation (Section 4.3). This integration equips the agent with zero-shot generalization ability over objects by grounding the novel object in a comprehensible visual representation.

4.1 Visual Grounding

Prior to segmentation, we must extract the correct target that the agent needs to interact with from the provided language instruction. Consider an example

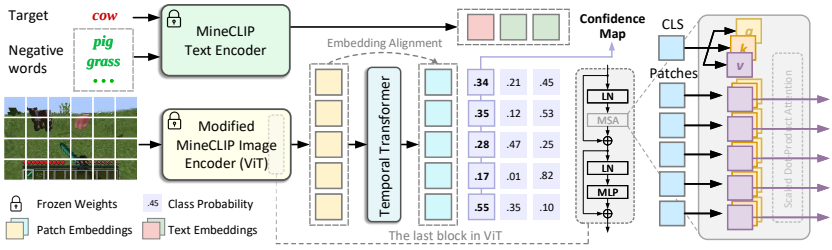


Fig. 2: Process of segmentation via MineCLIP. The modified MineCLIP image encoder takes as input the image and outputs patch embeddings, which are subsequently processed by the temporal transformer to guarantee embedding alignment. The MineCLIP text encoder encodes the target name along with a list of negative words. The probability of the target's presence on each patch is calculated based on the similarities between patch embeddings and text embeddings.

instruction: "`hunt a cow in plains with a diamond sword`". In this case, it is `cow` that should be extracted from the instruction as the target object, rather than `plains` or `diamond sword`, for the following segmentation. This can be easily accomplished by LLMs with appropriate prompts. Details on prompt designing and conversations with GPT-4 [1] can be found in Appendix A.1.

In the standard CLIP [42], the image encoder, a ResNet [20] or ViT [14], aggregates the visual features from all spatial locations through attention pooling. Recent works [29, 64] reveal that these features on each spatial location contain rich local information so that they can be used to perform zero-shot pixel-level predictions. In brief, the cosine similarities between these features and the outputs of the CLIP text encoder are also valid and informative. Concretely, MaskCLIP [64] makes use of the value-embedding of each spatial location in the last attention module, while CLIPSurgery [29] studies the feature of each spatial location in the final output and introduces an additional path. Inspired by MaskCLIP, we make adaptations to MineCLIP architecture to generate a confidence map for a specified target *without fine-tuning*.

To begin, we introduce the modification to the vision pathway of MineCLIP. We make changes to extract dense features from the last block of ViT. As illustrated in the rightmost part of Figure 2, the scaled dot-product attention in multi-head attention [53] module is removed, while the *value-embedding transformation* is retained. Then the transformed embeddings excluding that of `CLS` token are fed into the remaining modules within the ViT to obtain the final embedding of each patch. In this way, these patch embeddings share the same space as the original ViT output. As shown in Figure 2, the modified image encoder outputs patch embeddings instead of image embedding. However, these embeddings are not yet aligned with the embedding space of MineCLIP. In MineCLIP, the image encoder is followed by a temporal transformer that aggregates the embeddings of 16 images. Therefore, these patch embeddings also need to pass through the temporal transformer to guarantee alignment. Notably,

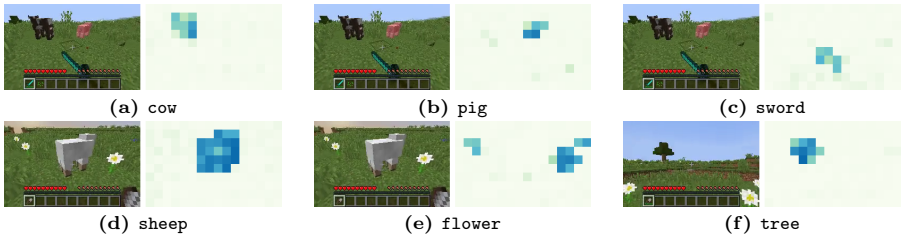


Fig. 3: Segmentation instances for targets: (a) `cow`, (b) `pig`, (c) `sword`, (d) `sheep`, (e) `flower`, and (f) `tree`. The darker blue the patch, the higher the probability of the target's presence on it.

these embeddings do not form a temporal sequence together as the input of the transformer. Instead, each patch embedding is individually processed by the temporal transformer, treated as a sequence of length 1. In this way, we obtain patch embeddings in the MineCLIP embedding space.

In the language pathway, no modification is made to the MineCLIP text encoder. The target name is encoded using the text encoder, along with a list of negative words [28, 64]. We construct a negative word list containing objects that frequently appear in Minecraft. For a detailed description of the word list, please refer to Appendix A.2. Given the patch embeddings encoded through the modified image encoder and the temporal transformer in the same embedding space of MineCLIP, we can calculate cosine similarities between patch embeddings and text embeddings, following the same approach as CLIP. Subsequently, we use softmax with the same temperature used in MineCLIP to determine the probabilities of objects' presence on each patch. Finally, we extract and reshape the probabilities of the target object to form the confidence map. The resulting confidence map consists of the same number of elements as the patches, with each element representing the probability of the target's presence on the corresponding patch. Examples of the confidence maps are shown in Figure 3.

In our preliminary experiment, we qualitatively attempt off-the-shelf open-vocabulary detection models [24, 32] but find their performance to be impaired by the domain gap between Minecraft and the real world, not as satisfactory as the results of our domain-specific model modified based on MineCLIP, as demonstrated in Appendix A.4. Another advantage of our method is that the generation of the confidence map can be integrated with the calculation of MineCLIP reward or the encoding of images using MineCLIP encoder, thus avoiding significant computational costs from incorporating additional segmentation models.

4.2 Transfer via Reward

The object-grounded confidence map of the target contains rich spatial information that can be utilized to facilitate RL through reward designing. The area occupied by the target in the image can serve as a *proxy* for estimating the distance to the target, based on the principle that the closer the target is to

the agent, the larger its area in the image and vice versa. Therefore, a reward proportional to the area of the target would guide the agent towards the target effectively. Additionally, we argue that the agent should be encouraged to aim at the target, *i.e.*, adjust the perspective to center the target in the field of view. This would help the agent further stabilize its orientation and increase the chance of interacting with the target when it is close enough. In Minecraft, interaction can only occur when the crosshair in the center of the agent view aligns with the target. Moreover, when multiple target objects are present in the view, the agent should learn to focus on a single target rather than attempting to keep all of them in view. This could also be interpreted in a more general way, such as humans usually place the target at the center of the visual field for better perception and interaction.

Based on these principles, we introduce an object-grounded intrinsic reward function named *focal* reward. At each time step t, it is computed as the mean of the Hadamard product between the current target confidence map m_t^c, and a Gaussian kernel denoted as m^k:

$$r_t^f = \text{mean}\left(m_t^c \circ m^k\right). \tag{1}$$

Here, m_t^c and m^k share the same dimensions with height H and width W. Each element of the Gaussian kernel is defined as:

$$m_{i,j}^k = \exp\left(-\frac{(i-\mu_1)^2}{2\sigma_1^2} - \frac{(j-\mu_2)^2}{2\sigma_2^2}\right), \tag{2}$$

where $\mu_1 = \frac{H+1}{2}$, $\sigma_1 = \frac{H}{3}$, $\mu_2 = \frac{W+1}{2}$, and $\sigma_2 = \frac{W}{3}$. This reward is designed to be proportional to the area occupied by the target and inversely proportional to the distance between the target patches and the center of the view.

As noted in [7], the MineCLIP reward, which relies on the similarity between the agent's preceding image observations and the provided instruction, is uncorrelated with the distance between the agent and the target. This phenomenon is demonstrated in Figure 4, where the MineCLIP reward does not consistently increase as the agent gets closer to the target. Consequently, in practice, the agent trained with the MineCLIP reward tends to "stare at" the target at a distance, rather than approaching it. This tendency obstructs the agent from learning some hard-exploration skills, particularly those that require multiple times of interactions with the targets, such as hunting. In contrast, our focal reward addresses this deficiency, increasing consistently when the agent approaches the target `cow`, as illustrated in Figure 4.

The confidence map generated from the modified MineCLIP may sometimes contain noisy activation [29, 64]. Therefore, we process the raw confidence map to enhance its quality before using it to compute the intrinsic reward. Firstly, we set the value corresponding to the patch where a word from the negative word list has the highest probability instead of the target, to zero. This operation reduces the impact of noisy activation on non-target patches. Secondly, we set values in the confidence map lower than a threshold $\tau = 0.2$ to zero, while those higher than this threshold are set to one, so as to amplify the distinction

(a) r^{mc} : 0.856, r^{f} : 0.029 **(b)** r^{mc} : 0.574, r^{f} : 0.046 **(c)** r^{mc} : 0.600, r^{f} : 0.071

Fig. 4: Comparison between MineCLIP reward r^{mc} and focal reward r^{f} at Frame 25, 35, and 45 in one episode of task `milk a cow`. From (a) to (c), r^{f} consistently increases as the agent approaches the target `cow`, while r^{mc} varies in an uncorrelated way.

between patches corresponding to the target and those unrelated to it. We ablate the Gaussian kernel and denoising process in Section 5.1.

4.3 Transfer via Representation

To train an instruction-following agent, the conventional practice involves directly taking the natural language instruction as the task representation into the policy network [15,21,23,38]. These instructions are typically encoded using a recurrent network or a language model such as BERT [22] and CLIP [42]. In contrast, we extract the target object from the instruction using GPT-4 [1] and subsequently convert it into a two-dimensional matrix, *i.e.*, the confidence map. Our underlying assumption is that this two-dimensional object-grounded representation offers more intuitive and accessible information for the policy network compared to the intricate space of language embeddings. When facing an instruction containing the name of an target object not encountered during training, our method grounds this novel text in the two-dimensional map, rendering it a comprehensible representation for the policy network. As a result, the agent can follow the guidance of the confidence map, navigate towards the novel target object, and finally interact with it.

In our implementation, we adopt the network architecture of MineAgent [17], which uses the MineCLIP image encoder to process image observations and MLPs to encode other information such as pose. We introduce an additional branch to encode the confidence map and fuse these features through concatenation. The policy network takes this fused multi-modality feature as input and outputs action distribution. Details regarding the policy network's architecture are available in Appendix B.2. We use PPO [45] as the base RL algorithm and train the agent with reward $r_t = r_t^{env} + \lambda r_t^{f}$, where r^{env} denotes the environmental reward and λ is a hyperparameter controlling the weight of the focal reward. According to the empirical results in Appendix E.3, we simply set $\lambda = 5$ for all experiments in the paper as we do not want to bother tuning this hyperparameter. We employ the multi-task RL paradigm, where the agent is trained to finish tasks in a predefined instruction set. Unlike typical multi-task reinforcement learning, our agent's learning objective is to not only master the training tasks but also to understand the mapping between the confidence map and the target object within the image observation, in order to perform zero-shot generalization to novel instructions involving unseen target objects.

Table 1: Success rates (%) of single-task RL with different reward functions on four challenging Minecraft tasks. Our focal reward enables RL to master all tasks by guiding the agent to consistently approach the target, while other baselines fail.

Tasks	Focal	MineCLIP	ND_{CLIP}	Sparse
`hunt a cow`	**71.3±9.7**	3.8±4.8	3.5±3.0	0.0±0.0
`hunt a sheep`	**68.8±25.3**	5.3±2.9	28.8±23.0	2.5±3.0
`hunt a pig`	**58.3±7.8**	2.3±1.7	0.3±0.5	0.5±0.6
`hunt a chicken`	**29.5±10.9**	0.0±0.0	4.8±1.5	0.5±0.6

5 Experiments

We conduct experiments in MineDojo [17], a Minecraft simulator that offers diverse open-ended tasks. Firstly, we perform single-task experiments to evaluate the effectiveness of our proposed focal reward. Then we extend our evaluation to multi-task experiments and examine the performance of COPL on multiple instructions. Lastly, but most importantly, we investigate the zero-shot generalization ability of COPL when confronted with instructions containing unseen targets. Details about Minecraft environments and RL hyperparameters in our experiments are described in Appendix B.1 and Appendix E.1, respectively.

5.1 Single-Task Experiments

Our single-task evaluation consists of tasks learning four challenging basic skills [3,7,17]: `hunt a cow`, `hunt a sheep`, `hunt a pig`, and `hunt a chicken`. In each task, the agent spawns in `plains` biome alongside several animals. The agent will receive a reward from the environment if it successfully kills the target animal. The difficulty of these basic skills lies in that animals, once attacked, will flee, requiring the agent to keep chasing and attacking the target animal. More details about the Minecraft task settings are available in Appendix C.1.

Evaluation. We compare our focal reward with the following three baselines: (1) **MineCLIP reward** [17] based on the similarity score between image observations and the instruction "`hunt a {animal} on plains with a diamond sword`"; (2) **ND_{CLIP} reward** [52], an intrinsic reward for exploration that measures the novelty of observation's MineCLIP embedding; (3) **Sparse reward**, training the agent with the environmental reward only. Results are reported in Table 1, including mean and variance, calculated from evaluating four models that are each trained with a unique random seed and the same number of environment steps (the same applies hereinafter). We can observe that only our focal reward leads to the mastery of all four skills by guiding the agent to consistently approach the target. In contrast, the MineCLIP reward fails because it cannot capture the distance between the agent and the target, offering limited benefit to these tasks. The failure of ND_{CLIP} reward suggests that exploration provides minimal assistance in learning these challenging skills due to the huge

Table 2: Success rates (%) of single-task RL with variants and ablation of our focal reward on two Minecraft tasks.

Tasks	Variants			Ablation	
	Focal	[raw]	[delta]	Focal	w/o kernel
`hunt a cow`	**71.3±9.7**	3.8±4.8	3.5±3.0	**67.0±15.0**	14.5±21.8
`hunt a sheep`	**68.8±25.3**	5.3±2.9	28.8±23.0	**82.3±3.4**	59.8±23.0

observation space of Minecraft. These methods' learning curves on each task are available in Appendix C.2. We also report results on harvest skill learning and additional analysis in Appendix C.3.

Variants and Ablation. To further investigate our focal reward, we compare it with two variants: Focal[raw], which uses the raw confidence map without denoising to compute the intrinsic reward, and Focal[delta], defined as $r_t^\delta = r_t^f - r_{t-1}^f$. The results in Table 2 demonstrate that our denoising process improves the effectiveness of the focal reward. We suppose that the poor performance of Focal[delta] may be linked to its sensitivity to segmentation noise, as it relies on differences in focal reward between two steps, making it susceptible to minor fluctuations in segmentation. In addition, we test the effectiveness of the Gaussian kernel, as presented in Table 2. We modify the environment settings to ensure that there are two target animals. The results prove the significance of the Gaussian kernel. Without this kernel, the reward may guide the agent to include both target animals in the view to acquire a high reward, hindering it from approaching either of them. In contrast, our focal reward addresses this problem by providing more reward in the center, thereby encouraging the agent to focus on a single target. A parameter study for the variance of the Gaussian kernel is available in Appendix E.3.

5.2 Multi-Task and Generalization Experiments

We conduct multi-task experiments to verify the effectiveness and zero-shot generalization capability of COPL. Given that tasks in Minecraft require different behavior patterns, we design two task domains, the **hunting domain** and the **harvest domain**. The hunting domain consists of four instructions: "`hunt a cow`", "`hunt a sheep`", "`hunt a pig`", and "`hunt a chicken`". These tasks share a common behavior pattern: repeatedly *approach the target*, *aim at it*, and *attack*. The harvest domain also contains four instructions: "`milk a cow`", "`shear a sheep`", "`harvest a flower`", and "`harvest leaves`". Tasks in the harvest domain are individually easier than those in the hunting domain but demand disparate behavior patterns. For example, "`harvest a flower`" requires the *attack* action while the other tasks require the *use* action. More details about the task settings in multi-task experiments are available in Appendix D.1.

Evaluation. We compare COPL with two language-conditioned reinforcement learning (LCRL) baselines, both taking language as input [21, 23, 33, 38]: (1)

Table 3: Success rates (%) of different multi-task RL methods in the hunting domain. (*upper*) Training hunting tasks. (*lower*) Test hunting tasks on **unseen** targets. COPL achieves 4× the performance of language-conditioned methods on unseen targets.

Tasks	COPL	LCRL[t]	LCRL[i]	One-Hot	[7]	STEVE-1
`cow`	**55.5±13.6**	18.8±22.2	12.8±8.5	23.0±20.3	31.0	6.0
`sheep`	**60.0±3.9**	31.8±25.2	24.3±17.7	31.0±12.7	30.0	14.0
`pig`	**50.5±18.4**	8.8±5.8	17.0±15.9	3.3±1.0	34.0	9.0
`chicken`	**34.0±3.2**	7.3±8.8	10.3±4.9	11.5±7.9	1.0	6.0
Avg.	**50.0±8.9**	16.6±13.3	16.1±8.8	17.2±7.9	24.0	8.8
`llama`	**48.8±6.5**	14.5±10.4	24.5±12.7	-	11.0	2.0
`horse`	**49.0±5.5**	2.5±1.3	5.5±4.7	-	7.0	3.0
`spider`	54.5±12.7	9.8±3.5	18.3±12.0	-	7.0	**73.0**
`mushroom`	**40.3±11.2**	19.3±20.5	0.0±0.0	-	26.0	4.0
Avg.	**48.1±6.6**	11.5±7.6	12.1±4.9	-	12.8	20.5

LCRL[t], which utilizes the target embedding encoded by the MineCLIP text encoder as input, sharing the same information as COPL albeit in language; (2) **LCRL[i]**, which utilizes the instruction embedding encoded by the MineCLIP text encoder as input. We also evaluate (3) **One-Hot**, a naive multi-task baseline, using a one-hot vector as the task indicator [3]. All these methods are trained with the focal reward and the only difference is their conditioning task representations. In the hunting domain, as shown in Table 3 (*upper*), COPL significantly outperforms other baselines, indicating that the confidence map provides a more accessible and informative task representation compared to the language embedding and one-hot vector, respectively. In contrast, the harvest domain presents a different picture. As illustrated in Table 4 (*upper*), all methods achieve similar performance. These results suggest that when tasks become easy enough, the impact of the task representation's complexity diminishes. These methods' learning curves on each task are available in Appendix D.2. We also evaluate two recent Minecraft basic skill models trained via imitation learning, [7] [4] and STEVE-1 [31]. Note that our intention *is not* to directly compare the performance of COPL with that of these two models, given the significant differences in their training paradigms and scopes of data. Rather, their evaluations only serve as a reference here due to the lack of alternative Minecraft foundation models trained via RL.

Generalization. Given that the two domains involve distinct behavior patterns, we conduct separate evaluations to assess the zero-shot generalization ability over

[3] One-Hot is not evaluated on unseen tasks due to the absence of corresponding one-hot vectors outside the training set.

[4] [7] is not evaluated in the harvest domain because the authors have not yet released the model trained for harvest tasks.

Table 4: Success rates (%) of different multi-task RL methods in the harvest domain. (*upper*) Training harvest tasks. (*lower*) Test harvest tasks on **unseen** targets. COPL achieves 2× the performance of language-conditioned methods on unseen targets.

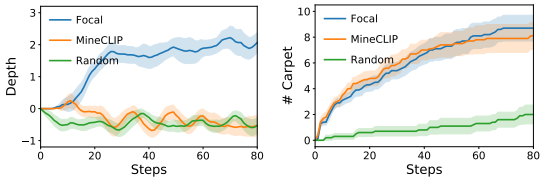
Tasks	COPL	LCRL[t]	LCRL[i]	One-Hot	STEVE-1
`milk`	**77.0±3.9**	68.0±4.6	73.5±11.7	69.5±5.0	2.0
`wool`	**55.5±5.7**	54.5±13.0	53.0±6.4	55.0±9.3	7.0
`flower`	85.5±7.7	**86.5±5.3**	81.5±6.5	80.3±6.8	43.0
`leaves`	**84.5±4.0**	83.7±2.5	73.7±5.9	74.0±5.9	66.0
Avg.	**75.6±3.2**	73.2±5.2	70.4±2.0	69.7±2.5	29.5
`water`	**46.5±12.7**	25.8±10.3	25.8±3.3	-	21.0
`mushroom`	**38.8±7.5**	29.0±6.2	31.3±3.1	-	0.0
`sand`	**24.0±8.4**	2.5±2.9	3.5±4.5	-	11.0
`dirt`	54.0±24.2	9.5±9.5	18.3±21.2	-	**75.0**
Avg.	**40.8±8.3**	16.7±4.3	19.7±6.3	-	26.8

target objects of these models trained in the hunting domain and the harvest domain. We test the hunting models with four novel instructions involving unseen animals: "`hunt a llama`", "`hunt a horse`", "`hunt a spider`", and "`hunt a mushroom cow`". The results in Table 3 (*lower*) show that COPL effectively transfers the learned skill to unseen targets, achieving high success rates. As we set the episode to be terminated if any animal is killed, the high success rates also prove COPL's ability to distinguish the target from other animals, rather than indiscriminately attacking them. Detailed precision of each method is reported in Appendix D.3. STEVE-1 shows poor performance across all hunting tasks except "`hunt a spider`". We suppose that its base model, VPT [3], possesses a strong prior on killing specific animals like spiders and heavily affects the behavior of STEVE-1 on following other hunting instructions. [7] achieves relatively higher success rates on "`hunt a cow`", "`hunt a sheep`", and "`hunt a pig`" due to these tasks being in its training set. Its lower performance on other tasks indicates its limitations in generalization ability.

For the harvest domain, we conduct evaluation on four unseen instructions, including "`harvest water`", "`shear a mushroom cow`", "`collect sand`", and "`collect dirt`". As shown in Table 4 (*lower*), COPL exhibits advantages on unseen tasks, reaching higher success rates than LCRL[t] and LCRL[i]. Compared to the hunting domain, results here more clearly demonstrate our method's superior generalization ability over target objects, as all methods perform at the same level on the training tasks. This indicates that better generalization ability emerges from grounding language in a visual representation. Evidenced by the precision reported in Appendix D.3, COPL recognizes unseen targets more accurately compared to LCRL. Similar to our observation in the hunting domain, STEVE-1 also possesses a strong tendency, which is digging in this case.

6 Discussion

Beyond Object-Centric Tasks. We focus on object-centric tasks in this paper, using an intrinsic reward to guide the agent towards the target. However, not all tasks in Minecraft necessitate approaching the target, such as the two Creative tasks, `dig a hole` and `lay the carpet`, adopted in [17]. We explore training separate single-task policies using the focal reward with targets, `hole` and `carpet`, respectively. Results in Figure 5 show its effective guidance towards task completion. Nevertheless, we observe that both our hunting and harvest models show limited performance on `dig a hole` due to its different behavior pattern from the training tasks and the difficulty in selecting a consistent target dirt block. Details on results of the two tasks are further discussed in Appendix G.



(a) `dig a hole` **(b)** `lay the carpet`

Fig. 5: (a) Digging depth and (b) number of laid carpets in one episode.

Limitations. One limitation of our method is the difficulty in defining a target for COPL to condition on, *i.e.*, for the agent to aim at, when facing non-object-centric tasks, such as `dig a hole` and `build a house`. Solving such creative and building tasks may entail generative models [62]. Additionally, our method compromises the *theoretical* generality of MineCLIP [17] due to not considering the action in instructions and only grounding target objects. Consequently, our method significantly improves its *practicality* in object-centric tasks, as verified in our experiments. Future work could focus on grounding language that describes actions and learning tasks requiring more complicated manipulation.

7 Conclusion

We propose COPL, a novel approach designed to address object-centric tasks and perform zero-shot object-level generalization in Minecraft. Our method effectively transfers the wealth of vision-language knowledge encoded in MineCLIP [17] into RL via reward designing and task representation. By comprehensive evaluations, we prove COPL's effectiveness in acquiring multiple basic skills and its generalization ability over target objects indicated in instructions, enabling it to apply the learned skills to follow unseen instructions that involve objects beyond the training tasks. Our work demonstrates the potential of integrating multimodal models, such as VLMs, into RL. Our method can be applied to other similar open-world environments by grounding natural language instructions in visual data and guiding the agent toward targets likewise. We hope COPL could contribute to the development of agents capable of understanding and responding to natural language instructions.

Acknowledgements

This work was supported by NSFC under grant 62250068. The authors would like to thank Ziluo Ding for providing help with the writing of this paper. The authors also would like to thank the anonymous reviewers for their valuable comments.

References

1. Achiam, J., Adler, S., Agarwal, S., Ahmad, L., Akkaya, I., Aleman, F.L., Almeida, D., Altenschmidt, J., Altman, S., Anadkat, S., et al.: Gpt-4 technical report. arXiv preprint arXiv:2303.08774 (2023)
2. Aubret, A., Matignon, L., Hassas, S.: A survey on intrinsic motivation in reinforcement learning. arXiv preprint arXiv:1908.06976 (2019)
3. Baker, B., Akkaya, I., Zhokov, P., Huizinga, J., Tang, J., Ecoffet, A., Houghton, B., Sampedro, R., Clune, J.: Video pretraining (vpt): Learning to act by watching unlabeled online videos. Advances in Neural Information Processing Systems **35**, 24639–24654 (2022)
4. Brohan, A., Brown, N., Carbajal, J., Chebotar, Y., Chen, X., Choromanski, K., Ding, T., Driess, D., Dubey, A., Finn, C., et al.: Rt-2: Vision-language-action models transfer web knowledge to robotic control. arXiv preprint arXiv:2307.15818 (2023)
5. Brohan, A., Brown, N., Carbajal, J., Chebotar, Y., Dabis, J., Finn, C., Gopalakrishnan, K., Hausman, K., Herzog, A., Hsu, J., et al.: Rt-1: Robotics transformer for real-world control at scale. arXiv preprint arXiv:2212.06817 (2022)
6. Bucker, A., Figueredo, L., Haddadin, S., Kapoor, A., Ma, S., Vemprala, S., Bonatti, R.: Latte: Language trajectory transformer. In: 2023 IEEE International Conference on Robotics and Automation (ICRA). pp. 7287–7294. IEEE (2023)
7. Cai, S., Wang, Z., Ma, X., Liu, A., Liang, Y.: Open-world multi-task control through goal-aware representation learning and adaptive horizon prediction. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 13734–13744 (2023)
8. Cai, S., Zhang, B., Wang, Z., Ma, X., Liu, A., Liang, Y.: Groot: Learning to follow instructions by watching gameplay videos. In: The Twelfth International Conference on Learning Representations (2023)
9. Chan, H., Wu, Y., Kiros, J., Fidler, S., Ba, J.: Actrce: Augmenting experience via teacher's advice for multi-goal reinforcement learning. arXiv preprint arXiv:1902.04546 (2019)
10. Chen, B., Xia, F., Ichter, B., Rao, K., Gopalakrishnan, K., Ryoo, M.S., Stone, A., Kappler, D.: Open-vocabulary queryable scene representations for real world planning. In: 2023 IEEE International Conference on Robotics and Automation (ICRA). pp. 11509–11522. IEEE (2023)
11. Colas, C., Karch, T., Lair, N., Dussoux, J.M., Moulin-Frier, C., Dominey, P., Oudeyer, P.Y.: Language as a cognitive tool to imagine goals in curiosity driven exploration. Advances in Neural Information Processing Systems **33**, 3761–3774 (2020)
12. Ding, Z., Wang, J., Tu, Z.: Open-vocabulary panoptic segmentation with maskclip. arXiv preprint arXiv:2208.08984 (2022)

13. Dorbala, V.S., Sigurdsson, G.A., Thomason, J., Piramuthu, R., Sukhatme, G.S.: Clip-nav: Using clip for zero-shot vision-and-language navigation. In: Workshop on Language and Robotics at CoRL 2022 (2022)
14. Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., et al.: An image is worth 16x16 words: Transformers for image recognition at scale. In: International Conference on Learning Representations (2020)
15. Du, Y., Watkins, O., Wang, Z., Colas, C., Darrell, T., Abbeel, P., Gupta, A., Andreas, J.: Guiding pretraining in reinforcement learning with large language models. arXiv preprint arXiv:2302.06692 (2023)
16. Ellis, B., Cook, J., Moalla, S., Samvelyan, M., Sun, M., Mahajan, A., Foerster, J., Whiteson, S.: Smacv2: An improved benchmark for cooperative multi-agent reinforcement learning. Advances in Neural Information Processing Systems **36** (2024)
17. Fan, L., Wang, G., Jiang, Y., Mandlekar, A., Yang, Y., Zhu, H., Tang, A., Huang, D.A., Zhu, Y., Anandkumar, A.: Minedojo: Building open-ended embodied agents with internet-scale knowledge. Advances in Neural Information Processing Systems **35**, 18343–18362 (2022)
18. Gadre, S.Y., Wortsman, M., Ilharco, G., Schmidt, L., Song, S.: Cows on pasture: Baselines and benchmarks for language-driven zero-shot object navigation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 23171–23181 (2023)
19. Gu, X., Lin, T.Y., Kuo, W., Cui, Y.: Open-vocabulary object detection via vision and language knowledge distillation. In: International Conference on Learning Representations (2021)
20. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 770–778 (2016)
21. Jiang, Y., Gu, S.S., Murphy, K.P., Finn, C.: Language as an abstraction for hierarchical deep reinforcement learning. Advances in Neural Information Processing Systems **32** (2019)
22. Kenton, J.D.M.W.C., Toutanova, L.K.: Bert: Pre-training of deep bidirectional transformers for language understanding. In: Proceedings of NAACL-HLT. pp. 4171–4186 (2019)
23. Khandelwal, A., Weihs, L., Mottaghi, R., Kembhavi, A.: Simple but effective: Clip embeddings for embodied ai. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 14829–14838 (2022)
24. Kirillov, A., Mintun, E., Ravi, N., Mao, H., Rolland, C., Gustafson, L., Xiao, T., Whitehead, S., Berg, A.C., Lo, W.Y., Dollár, P., Girshick, R.: Segment anything. arXiv:2304.02643 (2023)
25. Kirk, R., Zhang, A., Grefenstette, E., Rocktäschel, T.: A survey of zero-shot generalisation in deep reinforcement learning. Journal of Artificial Intelligence Research **76**, 201–264 (2023)
26. Kuo, W., Cui, Y., Gu, X., Piergiovanni, A., Angelova, A.: Open-vocabulary object detection upon frozen vision and language models. In: The Eleventh International Conference on Learning Representations (2022)
27. Kwon, M., Xie, S.M., Bullard, K., Sadigh, D.: Reward design with language models. In: The Eleventh International Conference on Learning Representations (2022)
28. Li, L.H., Zhang, P., Zhang, H., Yang, J., Li, C., Zhong, Y., Wang, L., Yuan, L., Zhang, L., Hwang, J.N., et al.: Grounded language-image pre-training. In: Proceed-

ings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 10965–10975 (2022)
29. Li, Y., Wang, H., Duan, Y., Li, X.: Clip surgery for better explainability with enhancement in open-vocabulary tasks. arXiv preprint arXiv:2304.05653 (2023)
30. Liang, F., Wu, B., Dai, X., Li, K., Zhao, Y., Zhang, H., Zhang, P., Vajda, P., Marculescu, D.: Open-vocabulary semantic segmentation with mask-adapted clip. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 7061–7070 (2023)
31. Lifshitz, S., Paster, K., Chan, H., Ba, J., McIlraith, S.: Steve-1: A generative model for text-to-behavior in minecraft. arXiv preprint arXiv:2306.00937 (2023)
32. Liu, S., Zeng, Z., Ren, T., Li, F., Zhang, H., Yang, J., Li, C., Yang, J., Su, H., Zhu, J., et al.: Grounding dino: Marrying dino with grounded pre-training for open-set object detection. arXiv preprint arXiv:2303.05499 (2023)
33. Luketina, J., Nardelli, N., Farquhar, G., Foerster, J., Andreas, J., Grefenstette, E., Whiteson, S., Rocktäschel, T.: A survey of reinforcement learning informed by natural language. arXiv preprint arXiv:1906.03926 (2019)
34. Mahmoudieh, P., Pathak, D., Darrell, T.: Zero-shot reward specification via grounded natural language. In: International Conference on Machine Learning. pp. 14743–14752. PMLR (2022)
35. Majumdar, A., Aggarwal, G., Devnani, B., Hoffman, J., Batra, D.: Zson: Zero-shot object-goal navigation using multimodal goal embeddings. Advances in Neural Information Processing Systems **35**, 32340–32352 (2022)
36. Mirchandani, S., Karamcheti, S., Sadigh, D.: Ella: Exploration through learned language abstraction. Advances in Neural Information Processing Systems **34**, 29529–29540 (2021)
37. Mnih, V., Kavukcuoglu, K., Silver, D., Graves, A., Antonoglou, I., Wierstra, D., Riedmiller, M.: Playing atari with deep reinforcement learning. arXiv preprint arXiv:1312.5602 (2013)
38. Mu, J., Zhong, V., Raileanu, R., Jiang, M., Goodman, N., Rocktäschel, T., Grefenstette, E.: Improving intrinsic exploration with language abstractions. Advances in Neural Information Processing Systems **35**, 33947–33960 (2022)
39. Nichol, A.Q., Dhariwal, P., Ramesh, A., Shyam, P., Mishkin, P., Mcgrew, B., Sutskever, I., Chen, M.: Glide: Towards photorealistic image generation and editing with text-guided diffusion models. In: International Conference on Machine Learning. pp. 16784–16804. PMLR (2022)
40. Nottingham, K., Ammanabrolu, P., Suhr, A., Choi, Y., Hajishirzi, H., Singh, S., Fox, R.: Do embodied agents dream of pixelated sheep?: Embodied decision making using language guided world modelling. arXiv preprint arXiv:2301.12050 (2023)
41. Oh, J., Singh, S., Lee, H., Kohli, P.: Zero-shot task generalization with multi-task deep reinforcement learning. In: International Conference on Machine Learning. pp. 2661–2670. PMLR (2017)
42. Radford, A., Kim, J.W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., et al.: Learning transferable visual models from natural language supervision. In: International conference on machine learning. pp. 8748–8763. PMLR (2021)
43. Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., Sutskever, I., et al.: Language models are unsupervised multitask learners. OpenAI blog **1**(8), 9 (2019)
44. Rao, Y., Zhao, W., Chen, G., Tang, Y., Zhu, Z., Huang, G., Zhou, J., Lu, J.: Denseclip: Language-guided dense prediction with context-aware prompting. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 18082–18091 (2022)

45. Schulman, J., Wolski, F., Dhariwal, P., Radford, A., Klimov, O.: Proximal policy optimization algorithms. arXiv preprint arXiv:1707.06347 (2017)
46. Selvaraju, R.R., Cogswell, M., Das, A., Vedantam, R., Parikh, D., Batra, D.: Grad-cam: Visual explanations from deep networks via gradient-based localization. In: Proceedings of the IEEE international conference on computer vision. pp. 618–626 (2017)
47. Shah, D., Osiński, B., Levine, S., et al.: Lm-nav: Robotic navigation with large pre-trained models of language, vision, and action. In: Conference on Robot Learning. pp. 492–504. PMLR (2023)
48. Shen, S., Li, L.H., Tan, H., Bansal, M., Rohrbach, A., Chang, K.W., Yao, Z., Keutzer, K.: How much can clip benefit vision-and-language tasks? arXiv preprint arXiv:2107.06383 (2021)
49. Shridhar, M., Manuelli, L., Fox, D.: Cliport: What and where pathways for robotic manipulation. In: Conference on Robot Learning. pp. 894–906. PMLR (2022)
50. Silver, D., Huang, A., Maddison, C.J., Guez, A., Sifre, L., Van Den Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M., et al.: Mastering the game of go with deep neural networks and tree search. nature **529**(7587), 484–489 (2016)
51. Stone, A., Xiao, T., Lu, Y., Gopalakrishnan, K., Lee, K.H., Vuong, Q., Wohlhart, P., Zitkovich, B., Xia, F., Finn, C., et al.: Open-world object manipulation using pre-trained vision-language models. arXiv preprint arXiv:2303.00905 (2023)
52. Tam, A., Rabinowitz, N., Lampinen, A., Roy, N.A., Chan, S., Strouse, D., Wang, J., Banino, A., Hill, F.: Semantic exploration from language abstractions and pre-trained representations. Advances in Neural Information Processing Systems **35**, 25377–25389 (2022)
53. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I.: Attention is all you need. Advances in neural information processing systems **30** (2017)
54. Vinyals, O., Babuschkin, I., Czarnecki, W.M., Mathieu, M., Dudzik, A., Chung, J., Choi, D.H., Powell, R., Ewalds, T., Georgiev, P., et al.: Grandmaster level in starcraft ii using multi-agent reinforcement learning. Nature **575**(7782), 350–354 (2019)
55. Wang, G., Xie, Y., Jiang, Y., Mandlekar, A., Xiao, C., Zhu, Y., Fan, L., Anandkumar, A.: Voyager: An open-ended embodied agent with large language models. arXiv preprint arXiv:2305.16291 (2023)
56. Wang, R., Mao, J., Hsu, J., Zhao, H., Wu, J., Gao, Y.: Programmatically grounded, compositionally generalizable robotic manipulation. In: The Eleventh International Conference on Learning Representations (2022)
57. Wang, Z., Cai, S., Liu, A., Jin, Y., Hou, J., Zhang, B., Lin, H., He, Z., Zheng, Z., Yang, Y., et al.: Jarvis-1: Open-world multi-task agents with memory-augmented multimodal language models. arXiv preprint arXiv:2311.05997 (2023)
58. Wang, Z., Cai, S., Liu, A., Ma, X., Liang, Y.: Describe, explain, plan and select: Interactive planning with large language models enables open-world multi-task agents. arXiv preprint arXiv:2302.01560 (2023)
59. Wu, J., Li, X., Yuan, S.X.H., Ding, H., Yang, Y., Li, X., Zhang, J., Tong, Y., Jiang, X., Ghanem, B., et al.: Towards open vocabulary learning: A survey. arXiv preprint arXiv:2306.15880 (2023)
60. Yuan, H., Zhang, C., Wang, H., Xie, F., Cai, P., Dong, H., Lu, Z.: Plan4mc: Skill reinforcement learning and planning for open-world minecraft tasks. arXiv preprint arXiv:2303.16563 (2023)

61. Zang, Y., Li, W., Zhou, K., Huang, C., Loy, C.C.: Open-vocabulary detr with conditional matching. In: European Conference on Computer Vision. pp. 106–122. Springer (2022)
62. Zhang, C., Cai, P., Fu, Y., Yuan, H., Lu, Z.: Creative agents: Empowering agents with imagination for creative tasks. arXiv preprint arXiv:2312.02519 (2023)
63. Zhang, T., Hu, Y., Cui, H., Zhao, H., Gao, Y.: A universal semantic-geometric representation for robotic manipulation. arXiv preprint arXiv:2306.10474 (2023)
64. Zhou, C., Loy, C.C., Dai, B.: Extract free dense labels from clip. In: European Conference on Computer Vision. pp. 696–712. Springer (2022)
65. Zhu, X., Chen, Y., Tian, H., Tao, C., Su, W., Yang, C., Huang, G., Li, B., Lu, L., Wang, X., et al.: Ghost in the minecraft: Generally capable agents for open-world enviroments via large language models with text-based knowledge and memory. arXiv preprint arXiv:2305.17144 (2023)