Lifelong Language-Conditioned Robotic Manipulation Learning

Xudong Wang[1,2*], Zebin Han[3,1,4*], Zhiyu Liu[1,2], Gan Li[3,1], Jiahua Dong[5], Baichen Liu[1], Lianqing Liu[1], Zhi Han[1†]

[1]State Key Laboratory of Robotics and Intelligent Systems, Shenyang Institute of Automation, Chinese Academy of Sciences
[2]University of Chinese Academy of Sciences
[3]North University of China
[4]Southeast University
[5]Mohamed bin Zayed University of Artificial Intelligence
{wangxudong,liuzhiyu,liubaichen,lqliu,hanzhi}@sia.cn.com, {zebinhan04,dongjiahua1995,gannonligan}@gmail.com

Abstract

Traditional language-conditioned manipulation agent sequential adaptation to new manipulation skills leads to catastrophic forgetting of old skills, limiting dynamic scene practical deployment. In this paper, we propose *SkillsCrafter*, a novel robotic manipulation framework designed to continually learn multiple skills while reducing catastrophic forgetting of old skills. Specifically, we propose a *Manipulation Skills Adaptation* to retain the old skills knowledge while inheriting the shared knowledge between new and old skills to facilitate learning of new skills. Meanwhile, we perform the singular value decomposition on the diverse skill instructions to obtain common skill semantic subspace projection matrices, thereby recording the essential semantic space of skills. To achieve forget-less and generalization manipulation, we propose a *Skills Specialization Aggregation* to compute inter-skills similarity in skill semantic subspaces, achieving aggregation of the previously learned skill knowledge for any new or unknown skill. Extensive experiments demonstrate the effectiveness and superiority of our proposed SkillsCrafter.

Code — https://skillscrafter-lifelong.github.io/

Introduction

Robotic manipulation has seen significant advancements facilitated by large vision-language models, enabling robots to interpret and execute complex instructions grounded in visual perception (Zhang et al. 2025c; Liu et al. 2025b; Zheng et al. 2025). In particular, language-conditioned robotic manipulation (LCRM) has emerged as a powerful paradigm that allows robots to follow open-ended natural language instructions. Recent manipulation works such as LH-VLA (Yao et al. 2025a), SRT-H (Kim et al. 2025), and PaLM-E (Driess, Jain et al. 2023) show impressive performance on LCRM in simulated and real-world environments.

Despite these advances, enabling robots to continually acquire new skills without forgetting previously learned skills remains a major challenge (Zahra et al. 2025). Most existing manipulation agents require task-specific fine-tuning, which leads to catastrophic forgetting of old skills when adapting to new skills (Wang et al. 2024a). This severely hinders the practical deployment of manipulation robots in dynamic environments where new skills are integrated over time without compromising existing competencies. As shown in Figure 1 (b), naively sequential fine-tuning an agent on skills results in a significant catastrophic forgetting on learned skills.

To address this problem, a straightforward method is to store separate models with full parameter checkpoints for each skill, which incurs substantial memory and computational overhead, making them impractical for long-term deployment. Recent large models leverage LoRA (Hu et al. 2022), a parameter-efficient tuning technique that injects small trainable low-rank matrices into a frozen base model. This fine-tuning method reduces storage by only maintaining lightweight LoRA adapters per skill task. However, it often treats each skill independently, ignoring the skill-shared knowledge (*e.g.*, basic actions like grasping, rotating, and pressing) across related manipulation skills. Moreover, selecting and loading the corresponding adapter at inference time typically requires manual intervention, which limits scalability. MoE-LoRA (Luo et al. 2024), HydraLoRA (Tian et al. 2024), Branch-LoRA (Zhang et al. 2025a), *et al.*, introduce an automatic routing between multiple LoRA experts via mixture-of-experts gating. However, these methods require pre-defining the number of experts, which risks overfitting or under-utilization when the number is suboptimal. These limitation highlights a critical need for more effective methods to continue robotic manipulation skills acquisition.

To handle the above practical scenarios, we introduce a new practical task named *Lifelong Language-Conditioned Robotic Manipulation (LLCRM)*. In this LLCRM setting, as shown in Figure 1 (a), a manipulation robotic agent continuously upgrades and evolves based on lifelong learning with multi-skills. LLCRM aims to enable agents to accumulate, reuse, and refine manipulation skills over time. Inspired by *how humans leverage analogies and prior experiences when acquiring new skills, we argue that efficient lifelong manipulation learning requires both skills* ***inheritance*** *and* ***integration***. Thus we identify two core challenges in LLCRM:

1. How to explore and exploit the shared-knowledge and

[*]These authors contributed equally.
[†]Corresponding Author

Copyright © 2026, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.

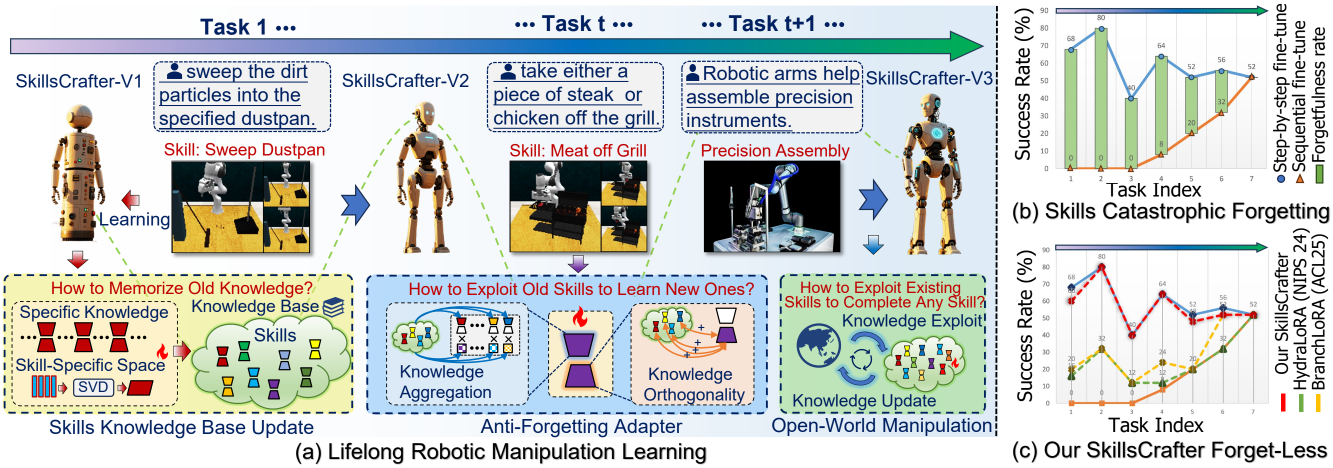

Figure 1: The proposed Lifelong Robotic Manipulation task. (a) SkillsCrafter is able to evolve and learn new skills based on learned skills. It maintains a skill knowledge base that stores learned knowledge to allow for efficient learning of new skills using shared knowledge of old skills, and for performing any unknown skill in the open world using previously learned knowledge. (b) Skills catastrophic forgetting under lifelong learning settings. (c) Our SkillsCrafter has a better anti-forgetting performance.

specific-knowledge between manipulation skills?

2. How to adaptively and flexibly aggregate the acquired knowledge for dealing with new or unknown skills?

To resolve the challenges in the LLCRM task, we propose SkillsCrafter, a novel lifelong manipulation framework that explicitly reuses inter-skill knowledge. SkillsCrafter maintains a skill knowledge base that stores learned skill knowledge to allow for efficient learning of new skills using previously learned skill knowledge, even for performing unknown skills in the open world. It comprises two core components: i). Manipulation Skills Adaptation (MSkA), which continually learns new skills while preserving old ones, by separating LoRA into skill-shared and skill-specific components with skills-shared knowledge inheritance and orthogonal constraints; ii). Skills Specialization Aggregation (SkSA), which computes inter-skills similarity in skill semantic subspace to achieve aggregation of the previously learned skills knowledge. And we also analyze the efficiency of our skills-shared knowledge inheritance strategy from theory. Extensive experiments in both simulated and real-world deployments have demonstrated that SkillsCrafter enables robots to learn new skills without forgetting previous ones, achieving the efficient SOTA performance in lifelong manipulation. The main contributions are outlined below:

- We introduce Lifelong Language-Conditioned Robotic Manipulation, a new task that enables agents to evolve and learn new skills based on learned skills. We also provide a new lifelong robotic manipulation benchmark for training and evaluating the lifelong manipulation task.
- To achieve efficient skills lifelong learning, we propose a Manipulation Skills Adaptation to retain old skills knowledge while inheriting the shared knowledge between new and old skills, to facilitate new skills learning.
- For anti-forgetting manipulation, we propose a Skills Subspaces Specialization Aggregation to compute inter-similarity in skill semantic subspaces, to achieve skills knowledge aggregation for manipulation skill inference.

Problem Formulation

Preliminary: In language-conditioned robotic manipulation task (Shridhar, Manuelli, and Fox 2023a; Team et al. 2024), an embodied agent is required to understand user language instruction $\mathcal{I}$ to complete a series of manipulation skills. Following LLARVA (Niu et al. 2024), we use a pre-trained LLM, i.e., LLaMA 2 (Touvron et al. 2023) with a tokenizer e, as the basic robotic agent $\mathcal{F}$. And to process the vision observation $\mathcal{O}$, we encode it using the CLIP vision-encoder $\mathcal{V}(\mathcal{O})$. The agent model architecture is similar to the multimodal large language model LLaVA (Liu et al. 2023). At each time i, the agent reasons the instruction $\mathcal{I}^i$ combined with the observation $\mathcal{O}^i$ to generate the next action as follows: $R^i = \mathcal{F}(\mathcal{V}(\mathcal{O}^i), e(\mathcal{I}^i))$. However, as shown in figure 1 (b), the agent's adaptation to a new skill S_t causes catastrophic forgetting of the old skills $\{S_1, S_2, ..., S_{t-1}\}$, which limits its flexible deployment, and the existing robotic manipulation agents struggle to grow in continual multi-skills.

Problem Definition: To address the above lifelong learning challenges, we introduce a new problem setting, Lifelong Language-Conditioned Robotic Manipulation (LLCRM). We define a multi-skills sequence set $\mathcal{S} = \{S_1, S_2, ..., S_t\}$, where t-th skill $S_t = \{\mathcal{O}_t, \mathcal{I}_t\}$ comprises vision observation $\mathcal{O}_t = \{o_1, o_2, ..., o_N\}$, and an instruction $\mathcal{I}_t$. Agent $\mathcal{F}$ is required to learn all the skills $\mathcal{S}$ sequentially, and all skills are tested after learning is complete. And in LLCRM settings, the skill-id t is agnostic during the testing phase. $\mathcal{I}_t$ of S_t does not overlap with any previous skills: $\mathcal{I}_t \bigcap (\bigcup_{j=1}^{t-1} \mathcal{I}_j) = \emptyset$. The LLCRM task aims to continually learn a sequence of new skills while alleviating the forgetting of old skills. In addition, to better explore and exploit skill knowledge, we define the *Skill Parameter Subspace* as the skill parameter learned by the agent to adapt to diverse

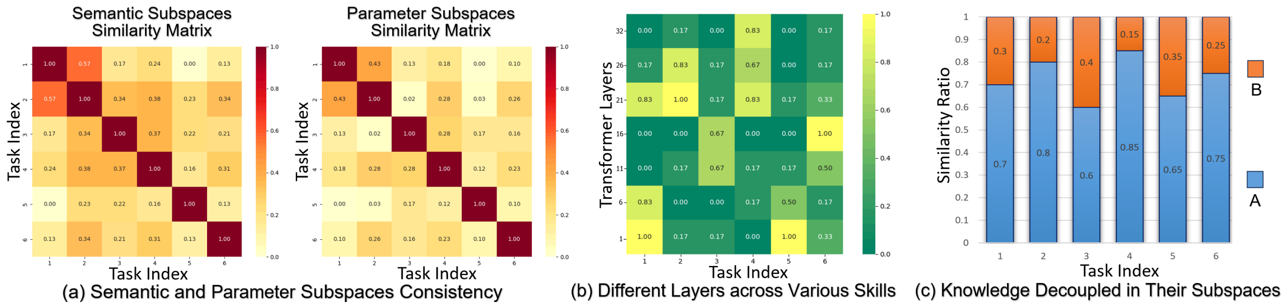

Figure 2: Illustration of the three sets of observation. (a) Subspaces Consistency: The semantic space can be used to associate with the parameter subspace. (b) Different Layers across Various Skills: The importance of different layers is not the same for different skills. (c) Knowledge Decoupled: **A** tends to learn shared knowledge, while **B** to learn specific knowledge, naturally.

new skills, the *Skill Semantic Subspace* as the semantic latent space embedding of user instructions describing skills.

Preliminary Study

In this section, we present our key observations regarding the two core challenges outlined in the Introduction. Various robotic skills not only have their specific knowledge, but also have shared knowledge. For example, "unscrewing a bottle cap" and "turning on a tap" both have rotating actions, and even "catching a bottle" and "roasting a chicken", which seem to be different skills, have commonalities in grasping actions. We use LoRA (Hu et al. 2022), $y = \mathbf{W}_0 x + \Delta\mathbf{W}x$, where $\Delta\mathbf{W}$ consists of two learnable matrices: $\mathbf{B} \in \mathbb{R}^{b\times 4}$, $\mathbf{A} \in \mathbb{R}^{4\times a}$, to fine-tune the pretrained LLARVA with six robotic skills, offering the following insightful observations.

Observation 1: ***Skill parameter subspaces and skill semantic spaces have similar correlations***. One straightforward idea is that if two skills are similar, the two parameter spaces for learning them also have similar correlations. We explore whether the semantic subspace and parameter subspace have similar correlations. Specifically, we calculate six skills' parameter subspace similarity matrices and semantic subspace similarity matrices. The parameter subspace similarity matrices measure the similarity between each of the skill parameter subspaces $\Delta\mathbf{W}$. The semantic subspace similarity matrices measure the similarity between each of the skill instruction semantic subspaces $E_T(\mathcal{I}_t)$ (CLIP text encoder $E_T(\cdot)$ (Radford et al. 2021)). We summarize the two similarity matrices in Figure 2 (a). We find that the two similarities are consistent in each skill task, which suggests that the instruction semantic space can be used to associate skill parameter subspaces. And in practice, for a new skill, its skill semantic subspace is explicit and easily obtainable, i.e., it can be obtained by encoding user manipulation instructions; while its skill parameter subspace is implicit and requires specific learning for obtaining. Therefore, this observation suggests we can estimate the new skill's parameter subspace from the similarity of the current semantic subspace with all previous semantic subspaces.

Observation 2: ***Skill subspaces required for different skills are distributed across different layers***. We also explore the distribution of learning skill spaces required for different skills across different transformer layers (from l-th to L-th layer). Specifically, we calculate the average value of each skill subspace in different layers $\{\Delta\bar{\mathbf{W}}_t\}_{l=1}^{L}$, as shown in Figure 2 (b). Based on this experimental result, we find that the distributions of learnable skill subspaces required for different skills across layers show significant differences, suggesting that the importance of different layers for different skills is also not the same. Therefore, to better learn specific knowledge for different skills, we should not assign the same fixed learning subspace for all manipulation skills.

Observation 3: ***Skill-shared and skill-specific knowledge are naturally decoupled in their subspaces***. We further explore the learning consistency of skill parameter subspaces for robotic manipulation multi-skills. Specifically, we calculate the similarity between each t-th skill subspace $\mathbf{B}_t, \mathbf{A}_t$ and the other five skills $\{\mathbf{B}_j, \mathbf{A}_j\}_{j\neq t}$, separately. We summarize their average values to obtain average similarity for each skill, as in Figure 2 (c). We find that the similarity of $\mathbf{A}_t$ is much greater than the similarity of $\mathbf{B}_t$, which suggests $\mathbf{A}_t$ tend to be consistent while $\mathbf{B}_t$ lack consistency. And a similar pattern was also reported by HydraLoRA (Tian et al. 2024) in natural language understanding and generation tasks. Therefore, this observation suggests that the skill subspace $\mathbf{A}$ tends to learn skill-shared knowledge, the subspace $\mathbf{B}$ tends to learn skill-specific knowledge, naturally.

The Proposed SkillsCrafter

The overall pipeline of SkillsCrafter is illustrated in Figure 3. It includes a Manipulation Skills Adaptation (MSkA) to learn skills continually, a Skills Specialization Aggregation (SkSA) to aggregate learned knowledge, and Skills Specified Inference (SkSI) to load knowledge for skill inference.

Manipulation Skills Adaptation

As shown in Figure 3(a), in order to learn the t-th skill S_t, we use the LoRA (Hu et al. 2022) to finetune the pretrained LLARVA $\mathcal{F}_{\theta_0}$ on skill-specific data $S_t = \{\mathcal{O}_t, \mathcal{I}_t\}$, and then obtain an updated model $\mathcal{F}'_{\theta'_t}$, where $\theta'_t = \theta_0 + \Delta\theta_t$, $\Delta\theta_t = \{\Delta\mathbf{W}_t^l\}_{l=1}^{L}$, and $\Delta\mathbf{W}_t^l = \mathbf{B}_t^l\mathbf{A}_t^l \in \mathbb{R}^{b_l\times a_l}$ is the updated low-rank weight in l-th layer of a total of L layers. $\mathbf{B}_t^l \in \mathbb{R}^{b_l\times r}$ and $\mathbf{A}_t^l \in \mathbb{R}^{r\times a_l}$ are the low-rank weights. To address LLCRM, a trivial solution is to train and save low-rank weights for all skills so far, and then load their specific weights at inference time. However, many skills share and

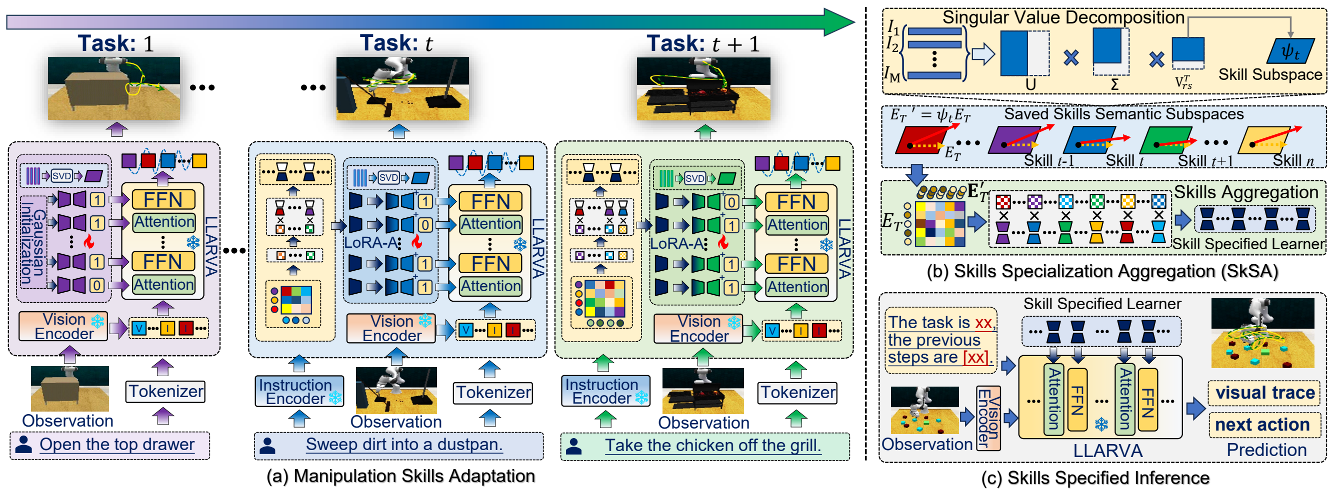

Figure 3: Illustration of the proposed SkillsCrafter pipeline. It includes (a) a *Manipulation Skills Adaptation* to achieve retaining the old skills knowledge while inheriting the shared knowledge between new and old skills to facilitate learning of new skills; (b) a *Skills Specialization Aggregation* to compute inter-skills similarity in skill-specific subspaces to achieve adaptive aggregation of skills knowledge; (c) a *Skills Specified Inference* to loads aggregated knowledge to achieve any skill manipulation inference.

have specific knowledge with each other; exploring and exploiting the knowledge promotes better new skills learning. For better skills adaptation, different from traditional LoRA, we use subspace $\mathbf{A}_t$ to learn skills-shared knowledge and $\mathbf{B}_t$ to learn skills-specific knowledge, based on ***Observation 3***.

Skills-Shared Knowledge: For skills-shared knowledge, we propose an adapter inheritance strategy to exploit the knowledge learned so far to promote better new skills learning. Specifically, in learning the initial skill S_1, we randomly initialize $\mathbf{A}_1^l$ and zero initialize the $\mathbf{B}_1^l$ matrix following standard LoRA (Hu et al. 2022). When learning subsequent new skills $S_t, t > 1$ continually, we perform knowledge aggregation and inheritance on the $\mathbf{A}_t^l$. We compute skills similarity as aggregation weight $\Omega_t = \{\omega_1, \omega_2, ...\omega_{t-1}\}$ within skill semantic subspace Ψ_t based on a SkSA Module (as Eq.(11)) to aggregate all the learned adapters $\mathcal{A}_t = \{\mathbf{A}_g^l\}_{g=1}^{t-1}$:

$$\mathbf{A}_{t*}^l = \Omega_t \odot \mathcal{A}_t, \quad \Omega_t = SkSA(\mathcal{I}_t, \Psi_{t-1}), \tag{1}$$

where $\Psi_{t-1} = \{\psi_1, \psi_2, ..., \psi_{t-1}\}$ are the saved skills semantic subspaces learned so far, and $\odot$ is the Hadamard product. The aggregated adapter $\mathbf{A}_{t*}^l$ is used as the initialization of $\mathbf{A}_t^l$ to inherit the skills-shared knowledge.

Skill-Special Knowledge: To further consolidate the skill shared-knowledge learning on skill parameter subspace $\mathbf{A}_t^l$ and keep the structure of shared knowledge stable, we also perform the skills orthogonal optimization on skill parameter subspace $\mathbf{B}_t^l$. Specifically, during adapting the t-th skill, we prefer that the skill-special parameter subspace $\mathbf{B}_t^l$ be orthogonal to the previous skill: $\sum_{i=1}^{t-1}\sum_{l=1}^{L}\mathrm{tr}((\mathbf{B}_i^l)^{\mathrm{T}}\mathbf{B}_t^l) = 0$. Furthermore, to prevent subspace $\mathbf{B}_t^l$ from degenerating into the trivial optimal solution, i.e., the zero matrix (without learning any knowledge), we perform L2 normalization on the subspace $\mathbf{B}_t^l$, i.e. project them onto the unit sphere:

$$R_t = \sum \mathrm{tr}((\widetilde{\mathbf{B}}_i^l)^{\mathrm{T}}\widetilde{\mathbf{B}}_t^l), \ \widetilde{\mathbf{B}}_i^l = \frac{\mathbf{B}_i^l}{||\,\mathbf{B}_i^l\,||_F + \epsilon}, \widetilde{\mathbf{B}}_t^l = \frac{\mathbf{B}_t^l}{||\,\mathbf{B}_t^l\,||_F + \epsilon}. \tag{2}$$

Thus the t-th skill adaptation loss for the LLM-based agent $\mathcal{F}$ performing auto-regressive action generation training is:

$$\mathcal{L}_t = -\sum_{n=1}^{N} logP_t(\hat{A}_{n:n+z-1}, \hat{\mathcal{P}}_{n:N|\mathcal{IO}_{t,n}}) + \lambda R_t, \tag{3}$$

where the $P_t(\hat{A}_{n:n+z-1}, \hat{\mathcal{P}}_{n:N|\mathcal{I}_t\mathcal{O}_{t,n}})$ denotes the predicted probability of ground-truth next z steps actions $\hat{A}_{n:n+z-1}$ and visual traces $\hat{\mathcal{P}}_{n:N}$ under the current observation $\mathcal{IO}_{t,n} = \{\mathcal{I}_t, \mathcal{O}_{t,n}\}$, and $z \le N$. Specifically, the auto-regressive generation predicted probability is calculated:

$$P_t(\hat{A}_{n:n+z-1}, \hat{\mathcal{P}}_{n:N|\mathcal{IO}_{t,n}}) = \prod_{i=n}^{n+z-1} P_{t,\Delta\theta_t}(x_i \mid \mathcal{IO}_{t,n}), \tag{4}$$

where $\Delta\theta_t$ are the all trainable parameters, and x_i is the current prediction token constituting manipulation actions.

In addition, due to the specificity of different skills, static loading of LoRA for fixed layers may lead to overfitting for simple skills (too many parameters) and underfitting for complex tasks (insufficient parameters). Based on ***Observation 2***, we propose a dynamic sparse LoRA loading strategy that adaptively decides whether to inject LoRA at each layer using a learnable Gumbel-Softmax gating mechanism. It uses Gumbel noise to approximate discrete choices as continuous distributions for enabling end-to-end training. Specifically, in each layer-wise LoRA $\Delta\mathbf{W}_t^l$, we add a learnable linear layer $H_{t,l}(\cdot)$ for each embedding $em_{t,l}$ to obtain $\mathbf{e}_{t,l} = H_{t,l}(em_{t,l}) \in \mathbb{R}^2$, where the two dimensions correspond to the options injection or no injection. We inject Gumbel noise $u_{t,l}^{(i)} = -\log(-\log(U_{t,l}^{(i)}))$, where $U \sim \mathrm{Uniform}(0, 1)$, compute Gumbel-Softmax distribution:

$$G_{t,l}^{(i)} = \frac{\exp((e_{t,l}^{(i)} + u_{t,l}^{(i)})/\tau_g)}{\sum_{j=1}^{2}\exp((e_{t,l}^{(j)} + u_{t,l}^{(j)})/\tau_g)}, \tag{5}$$

where τ_g is a temperature controlling decision discreteness. We use hard decision $g_t^l = \mathbb{I}(\arg\max_i G_{t,l}^{(i)} = 2)$ as forward propagation, and a soft gating $g_t^l = G_{t,l}^{(2)} \in [0,1]$ as backward propagation for training, to dynamically inject sparse LoRA:

$$\theta_t' = \theta_0 + g_t^l \cdot \Delta\mathbf{W}_t^l. \tag{6}$$

To encourage sparse injection, we add a sparsity regularization term $\mathcal{L}_s = \sum_{l=1}^{L} g_t^l$, and jointly optimize loss $\mathcal{L}_t + \lambda_s \mathcal{L}_s$.

Skills Specialization Aggregation

After learning a series of skills continually, SkillsCrafter requires recalling which of the learned knowledge is valuable for the current skill, and exploiting the knowledge to better complete the current skill. Based on ***Observation 1***, we can estimate the similarity of skill parameter subspaces from the similarity of skill semantic subspaces. Thus, we propose the Skills Specialization Aggregation (SkSA) module to adaptively aggregate the already learned adapters $\{\Delta\mathbf{W}_g\}_{g=1}^{t}$ based on skill semantic subspaces. Specifically, we use Singular Value Decomposition (SVD) (Lange and Lange 2010) to obtain a common skill semantic space for skills with the same skill but different instructions. Each skill semantic space is a discriminant subspace corresponding to its skill, and we perform similarity comparisons within these semantic subspaces to obtain skill similarity. The existing knowledge (parameter subspaces) is aggregated based on the similarity to achieve learning or inference of the current skill.

SVD for Common Skill Semantic Subspaces: When adapting new skills, we also calculate and store the common skill semantic spaces from diverse skill instructions. Specifically, as Figure 3(b), for each t-th skill, we use the CLIP text-encoder $E_T(\cdot)$ (Radford et al. 2021) to extract embeddings for diverse user instructions $X_t = [E_T(\mathcal{I}_t^1); E_T(\mathcal{I}_t^2); ...; E_T(\mathcal{I}_t^M)] \in \mathbb{R}^{M\times d}$, where M is the total number of instruction samples from t-th skill S_t and d is the extracted embedding dimension. These embeddings are then subjected to SVD to extract skill semantic subspace:

$$X_t = U_t \Sigma_t V_t^T, \tag{7}$$

where $U_t \in \mathbb{R}^{M\times M}$ is the left singular vector matrix, $\Sigma_t \in \mathbb{R}^{M\times d}$ is the singular values matrix and $V_t \in \mathbb{R}^{d\times d}$ is the right singular vector matrix. In order to maximize the distribution of the t-th common skill semantic subspace to reflect the distribution of the skill S_t, we select the first r_s rows of the right singular vector matrix V_t to represent the distribution of the embeddings of t-th skill, denoted $V_{t,r_s} = V_{[:,0:r_s]}$, thus we store orthogonal projection ψ_t for the S_t subspace:

$$\psi_t = V_{t,r_s} V_{t,r_s}^T, \tag{8}$$

where $\psi_t \in \mathbb{R}^{d\times d}$ is t-th skill semantic subspace projection.

Skill Specialization Aggregation: For new skill adapting or unknown skill inference, we adaptively aggregate all already learned knowledge $\{\Delta\mathbf{W}_g\}_{g=1}^{t}$ based on skill semantic subspaces $\Psi = \{\psi_1, \psi_2, ..., \psi_t\}$ according to any input instructions $\mathcal{I}_q$. Specifically, for skill S_q, we compute subspace projection $\mathbf{E}^{'}(\mathcal{I}_q)$ of $E_T(\mathcal{I}_q)$ onto every subspace Ψ:

$$\mathbf{E}^{'}(\mathcal{I}_q) = \{\psi_1 E_T(\mathcal{I}_q), \psi_2 E_T(\mathcal{I}_q), ..., \psi_t E_T(\mathcal{I}_q)\}, \tag{9}$$

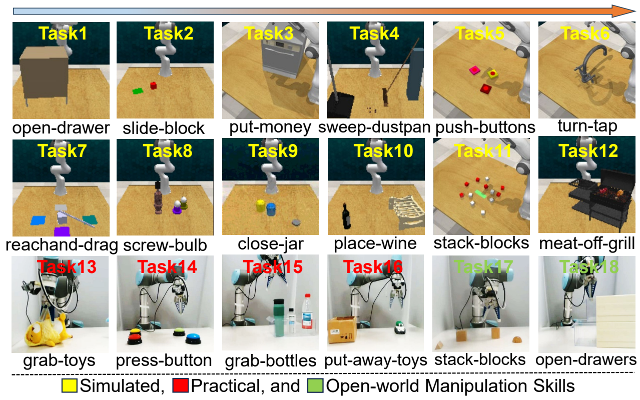

Figure 4: Illustration of the experiment robotic skill tasks setting. We establish a total of 12 robotic simulator skills and 6 real-environment robotic skills for incremental learning.

where $\mathbf{E}^{'}(\mathcal{I}_q)$ contains the projection of current known or unknown skill S_q onto the skill semantic subspaces Ψ. Subsequently, we calculate the angle between $E_T(\mathcal{I}_q)$ and every $\mathbf{E}_t^{'}(\mathcal{I}_q)$ with cosine similarity, which represents similarity degree of skill S_q on subspace ψ_t of learned skill S_t:

$$Sm_t(E_T(\mathcal{I}_q), \mathbf{E}_t^{'}(\mathcal{I}_q)) = \frac{E_T(\mathcal{I}_q)\mathbf{E}_t^{'}(\mathcal{I}_q)}{\parallel E_T(\mathcal{I}_q) \parallel \cdot \parallel \mathbf{E}_t^{'}(\mathcal{I}_q) \parallel}, \tag{10}$$

then, each cosine similarity Sm_t is transformed exponentially $(Sm_t)^\gamma$ and together constitutes the N weights Ω_q:

$$\Omega_q = \{\frac{(Sm_1)^\gamma}{\sum_{i=1}^{N}(Sm_i)^\gamma}, \frac{(Sm_2)^\gamma}{\sum_{i=1}^{N}(Sm_i)^\gamma}, ..., \frac{(Sm_N)^\gamma}{\sum_{i=1}^{N}(Sm_i)^\gamma}\}, \tag{11}$$

then we use Ω_q as weight to adaptively aggregate all the learned adapters $\{\Delta\mathbf{W}_t^l\}_{t=1}^{N}$ for obtaining aggregated $\Delta\tilde{\mathbf{W}}_q^l$:

$$\Delta\tilde{\mathbf{W}}_q^l = \Omega_q \odot \{\Delta\mathbf{W}_t^l\}_{t=1}^{N}. \tag{12}$$

The aggregated $\Delta\tilde{\mathbf{W}}_q^l$ adaptively aggregates previous similar knowledge to efficiently perform the current skill S_q.

Skills Specified Inference

After employing Eq.(12) to aggregate all low-rank adapters learned so far, SkillsCrafter loads the aggregated adapter to obtain an updated agent $\mathcal{F'}_{\theta_q'}$ specific to the current skill S_q. As Figure 3(c), SkillsCrafter loads the aggregated adapter to leverage learned knowledge, which can effectively mitigate the catastrophic forgetting of learned knowledge to better perform any known and unknown skill in the open world.

Experiments

Implementation Details

Manipulation Skills Benchmark Settings: To validate the proposed LLCRM task, we construct a comprehensive lifelong robotic manipulation benchmark. The benchmark settings are shown in Figure 4, including 12 common robotic manipulation skills based on RLBench simulator (James et al. 2020) and 6 real-world manipulation skills. The first 16

Methods	S1	S2	S3	S4	S5	S6	S7	S8	S9	S10	S11	S12	S13	S14	S15	S16	S17	S18	Avg.
Seq-FT (sequence fine-tune all)	0	0	0	0	0	0	0	0	0	0	0	0	12	4	4	80	8	8	6.4
LwF-LoRA (Li and Hoiem 2017)	0	4	0	4	0	4	0	0	0	0	0	0	8	12	0	84	4	0	6.7
EWC-LoRA (Xiang et al. 2023)	0	0	0	0	4	0	0	0	0	0	0	0	16	8	4	80	8	4	6.9
Dense MoLE (Chen et al. 2024)	4	24	4	12	8	8	8	0	4	0	0	8	24	20	28	80	8	8	13.8
Sparse MoLE (Dou et al. 2024)	16	32	8	8	12	4	8	4	4	8	0	4	28	16	24	76	4	4	14.4
MoLA (Gao et al. 2024)	12	20	12	8	16	8	12	4	0	4	4	12	44	24	24	80	0	8	16.2
HydraLoRA (Tian et al. 2024)	12	24	16	4	12	4	4	4	4	8	4	8	32	28	28	80	8	16	16.4
BranchLoRA (Zhang et al. 2025a)	16	28	8	12	8	12	16	0	8	4	0	8	40	36	32	76	12	12	18.2
O-LoRA (Wang et al. 2023a) + SkSA	56	72	**40**	60	48	52	36	4	12	8	4	**76**	84	84	76	**84**	36	52	49.1
SD-LoRA (Wu et al. 2025) + SkSA	52	80	32	64	44	44	**48**	4	20	**16**	8	72	88	84	80	80	36	48	50.0
SkillsCrafter (ours)	**56**	**80**	36	**64**	**48**	**52**	44	**12**	**20**	12	**12**	72	**92**	**84**	**80**	80	**40**	**52**	**52.0**

Table 1: Test results (Skill-Wise Average Success Rate, ASR ↑, %) of comparison experiment with LLCRM settings.

Methods	S1	S2	S3	S4	S5	S6	S7	S8	S9	S10	S11	S12	S13	S14	S15	S16	S17	S18	Avg.
Seq-FT	100	100	100	100	100	100	100	100	100	100	100	100	88	96	95	5	85	88	92.0
LwF-LoRA	100	95	100	95	100	93	100	100	100	100	100	100	92	87	100	0	92	100	91.9
EWC-LoRA	100	100	100	100	93	100	100	100	100	100	100	100	83	91	95	0	85	94	91.2
Dense MoLE	93	71	91	86	86	86	85	100	86	100	100	90	75	78	68	0	85	88	81.5
Sparse MoLE	73	62	82	90	79	93	85	80	86	50	100	95	71	83	73	0	92	94	77.0
MoLA	80	76	73	90	71	86	77	80	100	75	75	85	54	74	73	0	100	88	75.4
HydraLoRA	80	71	64	95	79	93	92	80	86	50	75	90	67	70	68	-5	85	76	73.1
BranchLoRA	73	67	82	86	86	79	69	100	71	75	100	90	58	61	64	0	77	82	73.3
O-LoRA + SkSA	7	14	**9**	29	14	7	31	80	57	50	75	**5**	13	9	14	0	31	24	25.9
SD-LoRA + SkSA	13	5	27	24	21	21	**8**	80	29	**0**	50	10	8	9	9	0	31	29	20.8
SkillsCrafter	**7**	**5**	18	**24**	**14**	**7**	15	**40**	**29**	25	**25**	10	**4**	**9**	**9**	0	**23**	**24**	**16.0**

Table 2: Test results (Skill-Wise Forgetting Rate, FR ↓, %) of comparison experiment with LLCRM settings.

skills serve as training and testing tasks for continual learning, while the last two skills are not trained and serve as open-world generalization testing. And our real-world manipulation platform is based on the UR-5 robotic arm with RGB camera. The agent is required to learn the 16 tasks in sequence, and then take a test on each skill after completing all the learning. The id t is agnostic during testing phase.

Training and Evaluation: For fair comparisons, our model and all comparison methods utilize LLARVA (Liu et al. 2023) as the baseline backbone. We use the Adam optimizer with an initial learning rate of 1.0×10^{-4} for training. All other hyperparameters are consistent with the LLARVA. We perform training and testing using PyTorch 2.1.2 with cu121 on eight NVIDIA RTX 6000 Ada Generation GPUs. Following the previous evaluation methods (Liu et al. 2023; Shridhar, Manuelli, and Fox 2023b; James et al. 2020), we perform each manipulation skill with 25 episodes, and we test the Average Success Rate (ASR) of each skill task:

$$ASR_t = \frac{1}{25}\sum_{n=1}^{25} SR_{t,n}, \tag{13}$$

where SR_t represents t-th skill completion status, SR_t is 1 when manipulation t-th skill is successfully executed, else, SR_t is 0. To evaluate agent continual learning ability, we also use FR_t to measure the t-th skill forgetting rate:

$$FR_t = \frac{\frac{1}{25}\sum_{n=1}^{25} SR_{gt,n} - ASR_t}{\frac{1}{25}\sum_{n=1}^{25} SR_{gt,n}}, \tag{14}$$

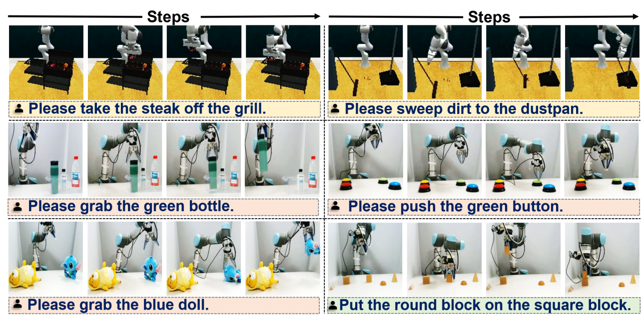

Figure 5: Illustration for some visualization examples.

where SR_{gt} represents the t-th skill completion status with learning only the 1-st to t-th skills. And SR_{gt} is 1 when t-th skill is successfully executed, else, SR_{gt} is 0. The larger FR_t, the greater the degree of t-th skill forgetting.

Comparison Experiment Results

This experiment demonstrates the superior robotic manipulation performance of our SkillsCrafter. Our comparison methods include the SOTA LoRA-based continual learning methods: **Seq-FT** is fine-tuned for all skills sequences; **LwF-LoRA** (Li and Hoiem 2017) uses knowledge distillation to retain previous skills performance; **EWC-LoRA** (Xiang et al. 2023) penalizes changes to critical parame-

Methods	GGM	SOT	INA	ASR %	FR %
Baseline	✗	✗	✗	5.8	92.8
SkillsCrafter w/o GGM	✗	✓	✓	46.9	26.5
SkillsCrafter w/o INA	✓	✓	✗	48.2	23.9
SkillsCrafter w/o SOT	✓	✗	✓	49.1	22.9
SkillsCrafter	✓	✓	✓	**52.0**	**16.0**

Table 3: Ablation study for skills shared/special knowledge.

Methods	IM	VM	AVG	TOP	ASR (%) ↑	FR (%)
SkillsCrafter w/ TOP	✗	✗	✗	✓	50.4	16.5
SkillsCrafter w/ Avg	✗	✗	✓	✗	44.4	26.3
SkillsCrafter w/ VM	✗	✓	✗	✗	48.9	20.2
SkillsCrafter	✓	✗	✗	✗	**52.0**	**16.0**

Table 4: Ablation study on skill knowledge aggregation.

ters of previous skills to mitigate forgetting; **Dense MoLE** (Chen et al. 2024) uses dense expert routing, while **Sparse MoLE** (Dou et al. 2024) uses sparse expert routing in MoE-LoRA; **MoLA** (Gao et al. 2024) introduces more experts at a deeper level based on Sparse MoLE; **O-LoRA** (Wang et al. 2023a) proposes orthogonal loss to learn task-specific knowledge; **HydraLoRA** (Tian et al. 2024) uses **A** to learn shared knowledge, and multiple **B** learn specific knowledge, **BranchLoRA** (Zhang et al. 2025a) further enhances the sparse selection mechanism; **SD-LoRA** (Wu et al. 2025) performs dynamic combination of previously learned skills LoRA. A summary of the evaluation results is shown in Table 1 and Table 2. Our SkillsCrafter achieves superior manipulation performance with an improved manipulation success rate (ASR: 52.0%, 2.0% improvement) and a lower forgetting rate (FR: 16.0%, 4.8% reduction). It also demonstrates superior performance on unknown skills (S17, S18), indicating better skill generalization. Some examples of the SkillsCrafter manipulation are provided in Figure 5.

Ablation Studies

How Do Skills Shared and Special Knowledge Perform? These ablation studies' results about skills shared and special knowledge are summarized in Table 3. Specifically, the "GGM" denotes the Gumbel-softmax Gating Mechanism based sparse loading strategy, which enhances the exploration of the skill-special knowledge; The "INA" denotes the Inheritance **A** strategy, which enhances the exploration of the skill-shared knowledge; The "SOT" denotes the Skills Orthogonal Term, which also enhances the exploration of the skill-special knowledge and consolidates the structure of LoRA's skill-shared and skill-special knowledge. Based on the ablation results, the proposed methods enable SkillsCrafter to explore and exploit skill-shared and special knowledge between diverse skills, effectively reducing catastrophic forgetting and improving new skill learning.

How Do Skills Knowledge Aggregation Perform? The ablation studies results about the knowledge aggregation methods are summarized in Table 4. Specifically, the "IM" denotes the Knowledge aggregation based on Instruction Matching; The "VM" denotes the Knowledge aggregation based on Vision Matching; The "Avg" denotes the average pooling for all instruction encodings, rather than using SVD; The "TOP" denotes the selection of the most similar LoRA in SkSA. Based on the ablation results, the proposed aggregation methods enable SkillsCrafter to exploit learned skill knowledge, effectively improving the LLCRM performance.

Related Works

Language-Conditioned Robotic Manipulation. Significant progress has been made in robotic manipulation, enabling agents to tackle increasingly complex tasks (Yao et al. 2025a; Jia et al. 2025; Huang et al. 2025; Yang et al. 2025; Li et al. 2025; Wang et al. 2024c, 2023b; Liang et al. 2024). For example, VIMA (Jiang et al. 2022) introduces a multi-modal prompting framework that formulates diverse manipulation tasks as a sequence modeling problem. Some methods (Xie et al. 2025; Goyal et al. 2023; Wang et al. 2024b) use voxelized 3D point cloud representations to better support sophisticated manipulations. PerAct (Shridhar, Manuelli, and Fox 2023b) improves manipulation performance by converting RGB-D observations into voxel grids and discretizing the action space. GNFactor (Ze et al. 2023) improves generalization by training the rendering of neural radiance fields. Despite advances, allowing robots to continually acquire skills without forgetting learned skills remains a challenge.

Continual Learning. Continual learning is essential for building adaptive AI systems that evolve over time (Wang et al. 2024a). Existing methods can be divided into three categories. Parameter regularization methods (Rebuffi et al. 2017; Li and Hoiem 2017; Derakhshani et al. 2021; Zhang et al. 2025b) constrain weight updates to preserve prior knowledge. Architecture-based methods (Liu et al. 2025a; Jung et al. 2020; Wu et al. 2021; Wang et al. 2022; Dong et al. 2025) dedicate specific network components to different tasks. Replay-based methods use either stored data (Bang et al. 2021; Sun et al. 2022) or generated samples (Li et al. 2022; Xiang et al. 2019; Sun et al. 2024) to mitigate forgetting. Several works explore applying continual learning in robotics (Meng et al. 2025; Yao et al. 2025b; Zhu et al. 2025). For example, LOTUS (Wan et al. 2024) enables continual learning by gradually expanding a skill library with new task demonstrations. However, these methods do not focus on skill-shared knowledge for better skills learning.

Conclusion

In this paper, we propose *SkillsCrafter*, a novel language-conditioned robotic manipulation model designed to continually learn multiple manipulation skills while reducing catastrophic forgetting of old skills. To explore shared and specific knowledge across multiple skills, we propose a *Manipulation Skills Adaptation* to inherit the skill knowledge between new and old skills to facilitate learning of new skills. We propose *Skills Specialization Aggregation* from the skill semantic space aggregation parameter space to achieve inference of any skills. Experiments in both simulation and real-world environments confirm its effectiveness.

Acknowledgments

This work was supported by the National Key Research and Development Program of China under Grant 2024YFB4707700, the National Natural Science Foundation of China under Grant T2596040, T2596045 and U23A20343, CAS Project for Young Scientists in Basic Research, Grant YSBR-041, Liaoning Provincial "Selecting the Best Candidates by Opening Competition Mechanism" Science and Technology Program under Grant 2023JH1/10400045, Fundamental Research Project of SIA under Grant 2024JC3K01.

References

Bang, J.; Kim, H.; Yoo, Y.; Ha, J.-W.; and Choi, J. 2021. Rainbow memory: Continual learning with a memory of diverse samples. In *CVPR*, 8218–8227.

Chen, C.; Zhu, J.; Luo, X.; Shen, H.; Song, J.; and Gao, L. 2024. Coin: A benchmark of continual instruction tuning for multimodel large language models. *NeurIPS*, 37: 57817–57840.

Derakhshani, M. M.; Zhen, X.; Shao, L.; and Snoek, C. 2021. Kernel continual learning. In *ICML*, 2621–2631.

Dong, J.; Wang, X.; Liang, W.; Han, Z.; Cao, M.; Zhang, D.; Zhao, H.; Han, Z.; Khan, S.; and Khan, F. S. 2025. Bring Your Dreams to Life: Continual Text-to-Video Customization. *arXiv preprint arXiv:2512.05802*.

Dou, S.; Zhou, E.; Liu, Y.; Gao, S.; Shen, W.; Xiong, L.; Zhou, Y.; Wang, X.; Xi, Z.; Fan, X.; et al. 2024. LoRAMoE: Alleviating world knowledge forgetting in large language models via MoE-style plugin. In *ACL*, 1932–1945.

Driess, D.; Jain, A.; et al. 2023. PaLM-E: An Embodied Multimodal Language Model. *arXiv preprint arXiv:2303.03378*.

Gao, C.; Chen, K.; Rao, J.; Sun, B.; Liu, R.; Peng, D.; Zhang, Y.; Guo, X.; Yang, J.; and Subrahmanian, V. 2024. Higher layers need more lora experts. *arXiv preprint arXiv:2402.08562*.

Goyal, A.; Xu, J.; Guo, Y.; Blukis, V.; Chao, Y.-W.; and Fox, D. 2023. Rvt: Robotic view transformer for 3d object manipulation. In *CORL*, 694–710.

Hu, E. J.; Shen, Y.; Wallis, P.; Allen-Zhu, Z.; Li, Y.; Wang, S.; Wang, L.; Chen, W.; et al. 2022. Lora: Low-rank adaptation of large language models. *ICLR*, 1(2): 3.

Huang, H.; Chen, X.; Chen, Y.; Li, H.; Han, X.; Wang, Z.; Wang, T.; Pang, J.; and Zhao, Z. 2025. RoboGround: Robotic Manipulation with Grounded Vision-Language Priors. arXiv:2504.21530.

James, S.; Ma, Z.; Arrojo, D. R.; and Davison, A. J. 2020. RLBench: The Robot Learning Benchmark & Learning Environment. *IEEE Robotics and Automation Letters*, 5: 3019–3026.

Jia, M.; Huang, H.; Zhang, Z.; Wang, C.; Zhao, L.; Wang, D.; Liu, J. X.; Walters, R.; Platt, R.; and Tellex, S. 2025. Learning Efficient and Robust Language-Conditioned Manipulation Using Textual-Visual Relevancy and Equivariant Language Mapping. *IEEE Robotics and Automation Letters*, 10(8): 8204–8211.

Jiang, Y.; Gupta, A.; Zhang, Z.; Wang, G.; Dou, Y.; Chen, Y.; Fei-Fei, L.; Anandkumar, A.; Zhu, Y.; and Fan, L. 2022. Vima: General robot manipulation with multimodal prompts. arXiv:2210.03094.

Jung, S.; Ahn, H.; Cha, S.; and Moon, T. 2020. Continual learning with node-importance based adaptive group sparse regularization. *NeurIPS*, 33: 3647–3658.

Kim, J. W. B.; Chen, J.-T.; Hansen, P.; Shi, L. X.; Goldenberg, A.; Schmidgall, S.; Scheikl, P. M.; Deguet, A.; White, B. M.; Tsai, D. R.; Cha, R. J.; Jopling, J.; Finn, C.; and Krieger, A. 2025. SRT-H: A hierarchical framework for autonomous surgery via language-conditioned imitation learning. *Science Robotics*, 10: eadt5254.

Lange, K.; and Lange, K. 2010. Singular value decomposition. *Numerical analysis for statisticians*, 129–142.

Li, G.; Zhai, Y.; Chen, Q.; Gao, X.; Zhang, J.; and Zhang, Y. 2022. Continual few-shot intent detection. In *COLING*, 333–343.

Li, X.; Xu, L.; Zhang, M.; Liu, J.; Shen, Y.; Ponomarenko, I.; Xu, J.; Heng, L.; Huang, S.; Zhang, S.; and Dong, H. 2025. CrayonRobo: Object-Centric Prompt-Driven Vision-Language-Action Model for Robotic Manipulation. arXiv:2505.02166.

Li, Z.; and Hoiem, D. 2017. Learning without forgetting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(12): 2935–2947.

Liang, W.; Sun, G.; He, Q.; Ren, Y.; Dong, J.; and Cong, Y. 2024. Never-Ending Behavior-Cloning Agent for Robotic Manipulation. arXiv:2403.00336.

Liu, B.; Lyu, Q.; Wang, X.; Dong, J.; Liu, L.; and Han, Z. 2025a. CRISP: Contrastive Residual Injection and Semantic Prompting for Continual Video Instance Segmentation. arXiv:2508.10432.

Liu, H.; Li, C.; Wu, Q.; and Lee, Y. J. 2023. Visual instruction tuning. *NeurIPS*, 36: 34892–34916.

Liu, Y.; Chen, W.; Bai, Y.; Liang, X.; Li, G.; Gao, W.; and Lin, L. 2025b. Aligning Cyber Space With Physical World: A Comprehensive Survey on Embodied AI. *IEEE/ASME Transactions on Mechatronics*, 1–22.

Luo, T.; Lei, J.; Lei, F.; Liu, W.; He, S.; Zhao, J.; and Liu, K. 2024. Moelora: Contrastive learning guided mixture of experts on parameter-efficient fine-tuning for large language models. *arXiv preprint arXiv:2402.12851*.

Meng, Y.; Bing, Z.; Yao, X.; Chen, K.; Huang, K.; Gao, Y.; Sun, F.; and Knoll, A. 2025. Preserving and combining knowledge in robotic lifelong reinforcement learning. *Nature Machine Intelligence*, 1–14.

Niu, D.; Sharma, Y.; Biamby, G.; Quenum, J.; Bai, Y.; Shi, B.; Darrell, T.; and Herzig, R. 2024. LLARVA: Vision-Action Instruction Tuning Enhances Robot Learning. In *CORL*.

Radford, A.; Kim, J. W.; Hallacy, C.; Ramesh, A.; Goh, G.; Agarwal, S.; Sastry, G.; Askell, A.; Mishkin, P.; Clark, J.; et al. 2021. Learning transferable visual models from natural language supervision. In *ICML*, 8748–8763.

Rebuffi, S.-A.; Kolesnikov, A.; Sperl, G.; and Lampert, C. H. 2017. icarl: Incremental classifier and representation learning. In *CVPR*, 2001–2010.

Shridhar, M.; Manuelli, L.; and Fox, D. 2023a. Perceiver-actor: A multi-task transformer for robotic manipulation. In *CORL*, 785–799.

Shridhar, M.; Manuelli, L.; and Fox, D. 2023b. Perceiver-actor: A multi-task transformer for robotic manipulation. In *CORL*, 785–799. PMLR.

Sun, G.; Liang, W.; Dong, J.; Li, J.; Ding, Z.; and Cong, Y. 2024. Create your world: Lifelong text-to-image diffusion. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Sun, Q.; Lyu, F.; Shang, F.; Feng, W.; and Wan, L. 2022. Exploring example influence in continual learning. *NeurIPS*, 35: 27075–27086.

Team, O. M.; Ghosh, D.; Walke, H.; Pertsch, K.; Black, K.; Mees, O.; Dasari, S.; Hejna, J.; Kreiman, T.; Xu, C.; et al. 2024. Octo: An open-source generalist robot policy. *arXiv preprint arXiv:2405.12213*.

Tian, C.; Shi, Z.; Guo, Z.; Li, L.; and Xu, C.-Z. 2024. Hydralora: An asymmetric lora architecture for efficient fine-tuning. *NeurIPS*, 37: 9565–9584.

Touvron, H.; Martin, L.; Stone, K.; Albert, P.; Almahairi, A.; Babaei, Y.; Bashlykov, N.; Batra, S.; Bhargava, P.; Bhosale, S.; et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Wan, W.; Zhu, Y.; Shah, R.; and Zhu, Y. 2024. Lotus: Continual imitation learning for robot manipulation through unsupervised skill discovery. In *ICRA*, 537–544.

Wang, L.; Zhang, X.; Li, Q.; Zhu, J.; and Zhong, Y. 2022. Coscl: Cooperation of small continual learners is stronger than a big one. In *ECCV*, 254–271.

Wang, L.; Zhang, X.; Su, H.; and Zhu, J. 2024a. A comprehensive survey of continual learning: Theory, method and application. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Wang, X.; Chen, T.; Ge, Q.; Xia, H.; Bao, R.; Zheng, R.; Zhang, Q.; Gui, T.; and Huang, X. 2023a. Orthogonal subspace learning for language model continual learning. *arXiv preprint arXiv:2310.14152*.

Wang, X.; Chen, X.; Ren, W.; Han, Z.; Fan, H.; Tang, Y.; and Liu, L. 2024b. Compensation Atmospheric Scattering Model and Two-Branch Network for Single Image Dehazing. *IEEE Transactions on Emerging Topics in Computational Intelligence*, 8(4): 2880–2896.

Wang, X.; Chen, X.; Wang, F.; Xu, C.; and Tang, Y. 2023b. Image Recovery and Object Detection Integrated Algorithms for Robots in Harsh Battlefield Environments. In *Intelligent Robotics and Applications*, 575–585.

Wang, X.; Ren, W.; Chen, X.; Fan, H.; Tang, Y.; and Han, Z. 2024c. Uni-YOLO: Vision-Language Model-Guided YOLO for Robust and Fast Universal Detection in the Open World. In *Proceedings of the 32nd ACM International Conference on Multimedia*, MM '24, 1991–2000.

Wu, Y.; Piao, H.; Huang, L.-K.; Wang, R.; Li, W.; Pfister, H.; Meng, D.; Ma, K.; and Wei, Y. 2025. S-LoRA: Scalable Low-Rank Adaptation for Class Incremental Learning. *arXiv preprint arXiv:2501.13198*.

Wu, Z.; Baek, C.; You, C.; and Ma, Y. 2021. Incremental learning via rate reduction. In *CVPR*, 1125–1133.

Xiang, J.; Tao, T.; Gu, Y.; Shu, T.; Wang, Z.; Yang, Z.; and Hu, Z. 2023. Language models meet world models: Embodied experiences enhance language models. *NeurIPS*, 36: 75392–75412.

Xiang, Y.; Fu, Y.; Ji, P.; and Huang, H. 2019. Incremental learning using conditional adversarial networks. In *ICCV*, 6619–6628.

Xie, J.; Zhou, F.; Liu, J.; and Wang, C. 2025. Semi-Supervised Language-Conditioned Grasping With Curriculum-Scheduled Augmentation and Geometric Consistency. *IEEE Robotics and Automation Letters*, 10: 4021–4028.

Yang, J.; Tan, W.; Jin, C.; Yao, K.; Liu, B.; Fu, J.; Song, R.; Wu, G.; and Wang, L. 2025. Transferring Foundation Models for Generalizable Robotic Manipulation. In *WACV*, 1999–2010.

Yao, X.; Blei, T.; Meng, Y.; Zhang, Y.; Zhou, H.; Bing, Z.; Huang, K.; Sun, F.; and Knoll, A. 2025a. Long-Horizon Language-Conditioned Imitation Learning for Robotic Manipulation. *IEEE/ASME Transactions on Mechatronics*, 1–12.

Yao, Y.; Liu, S.; Song, H.; Qu, D.; Chen, Q.; Ding, Y.; Zhao, B.; Wang, Z.; Li, X.; and Wang, D. 2025b. Think Small, Act Big: Primitive Prompt Learning for Lifelong Robot Manipulation. In *CVPR*, 22573–22583.

Zahra, A.; Irshad, M. J.; Wasif, M.; and Rizwan, M. 2025. State of the art in lifelong learning of robots. In *ICEPECC*, volume 2025, 458–463. IET.

Ze, Y.; Yan, G.; Wu, Y.-H.; Macaluso, A.; Ge, Y.; Ye, J.; Hansen, N.; Li, L. E.; and Wang, X. 2023. Gnfactor: Multi-task real robot learning with generalizable neural feature fields. In *CORL*, 284–301. PMLR.

Zhang, D.; Ren, Y.; Li, Z.-Z.; Yu, Y.; Dong, J.; Li, C.; Ji, Z.; and Bai, J. 2025a. Enhancing Multimodal Continual Instruction Tuning with BranchLoRA. *arXiv preprint arXiv:2506.02041*.

Zhang, J.; Cai, K.; Yang, J.; and Wang, K. 2025b. Learning Dynamics of VLM Finetuning. *arXiv preprint arXiv:2510.11978*.

Zhang, Q.; Liu, Z.; Fan, H.; Liu, G.; Zeng, B.; and Liu, S. 2025c. Flowpolicy: Enabling fast and robust 3d flow-based policy via consistency flow matching for robot manipulation. In *AAAI*, volume 39, 14754–14762.

Zheng, Y.; Yao, L.; Su, Y.; Zhang, Y.; Wang, Y.; Zhao, S.; Zhang, Y.; and Chau, L.-P. 2025. A survey of embodied learning for object-centric robotic manipulation. *Machine Intelligence Research*, 1–39.

Zhu, R.; Sun, E.; Huang, G.; and Celiktutan, O. 2025. Efficient Continual Adaptation of Pretrained Robotic Policy with Online Meta-Learned Adapters. *arXiv preprint arXiv:2503.18684*.