AutoMA: Automated Modular Attention enables Context-Rich Imitation Learning using Foundation Models

Yifan Zhou[1], Xiao Liu[1], Quan Vuong[2], and Heni Ben Amor[1]

***Abstract*— Although imitation learning offers an appealing framework for policy learning, scaling it is challenging due to the lack of high-quality expert data. Previous works have addressed this by modifying model design and increasing state diversity to improve data quality, but these approaches often overlook the rich contextual information inherent in the data itself. In this paper, we propose an automated learning framework that leverages large language models (LLMs) and vision-language models (VLMs) to provide context-rich supervision for Modular Attention based imitation learning policies, denoted as AutoMA. Specifically, LLMs are used for hierarchical modular design, and VLMs provide supervision signals for the different modules. AutoMA thus leverages the capabilities of foundational models to inform the policy with rich context from the data. Experimentally, we demonstrate that AutoMA is scalable across a broad spectrum of tasks and significantly outperforms baseline models in six simulated and real-world manipulation tasks, achieving success rate improvements of up to 63.3%. Furthermore, we illustrate how AutoMA facilitates re-using the modules when transferring the policy to different robots with only 20% of the original data scale, significantly improving data efficiency. The project page is at `https://auto-ma.github.io/`.**

I. INTRODUCTION

Imitation learning is vital in robotics and machine learning, allowing robots to learn skills by mimicking expert demonstrations [1]–[7]. It is preferred for its ability to teach complex tasks without extensive programming, making learning and adaptation more accessible [8]–[10]. However, obtaining large-scale expert demonstrations for imitation learning remains a significant challenge [11]. Beyond the labor-intensive and time-consuming data collection process, a common issue is the quality of the data, particularly whether the state and action pairs provide sufficient representation for the task [12], [13]. Several prior works have addressed this challenge by modifying the model with task-specific assumptions. These approaches include incorporating longer history information [14], conditioning on prior generated actions [15], injecting expert policy to guide the behavior of the student policy [16], and modifying perception or action representations [17]–[19]. However, these approaches overlook the potential of leveraging the context inherent in the data itself for policy learning, rather than just altering the algorithm. For example, as shown in Figure 1, the task "*stack the blue block on the red block*" contains a planning hierarchy within the language instruction, indicating that the blue block needs to be picked up first and placed on the red block. Additionally, observation images can help infer and locate the blocks. This context is often disregarded in conventional end-to-end imitation learning pipelines.

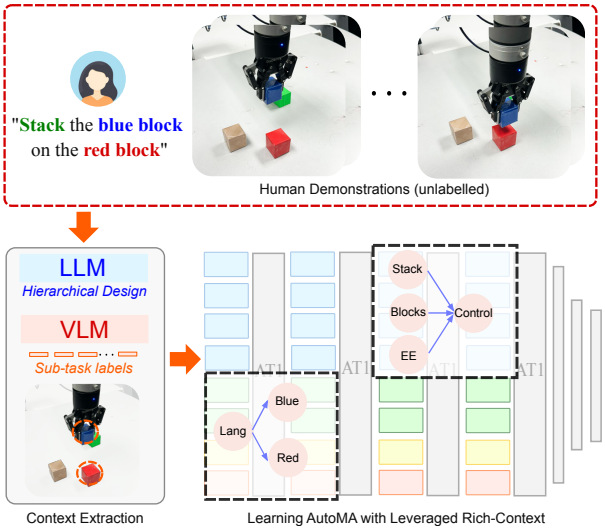

Fig. 1. Given the task "*stack the blue block on the red block*", LLMs design hierarchical modules h by decomposing the task into several sub-task modules, while VLMs supervises the module. The policy network, enriched with this context, can then effectively execute the task.

An effective approach to improve imitation learning policies should involve leveraging the useful context inherent in the data while simultaneously designing models with task-specific assumptions. One method to embed context within a model is through modularity [20], [21], allowing task-specific contexts to be learned within different sub-modules in parallel. While this approach enhances data efficiency and improves performance, it requires significant manual effort for labeling data to define the sub-modules correctly, which can be labor-intensive and time-consuming, especially in complex tasks.

In this paper, we address the challenges of labor-intensive data labeling and manual sub-module design by leveraging large language models (LLMs) and vision-language models (VLMs) to automatically design the sub-modules, their hierarchies, and extract their labels. This approach systematically integrates context extraction from data with task-specific architectural design, allowing for more efficient policy learning. This automated pipeline for policy learning is named **Auto**mated **M**odular **A**ttention (AutoMA). As shown in Figure 1, AutoMA utilizes the reasoning capabilities of large language models (LLMs) to decompose tasks into several sub-tasks, which are treated as modules. Vision-Language

[1]Authors are with the School of Computing and Augmented Intelligence, Arizona State University {yzhou298, xliu330, hbenamor}@asu.edu. [2]Author is with Physical Intelligence, {quan.hovuong}@gmail.com

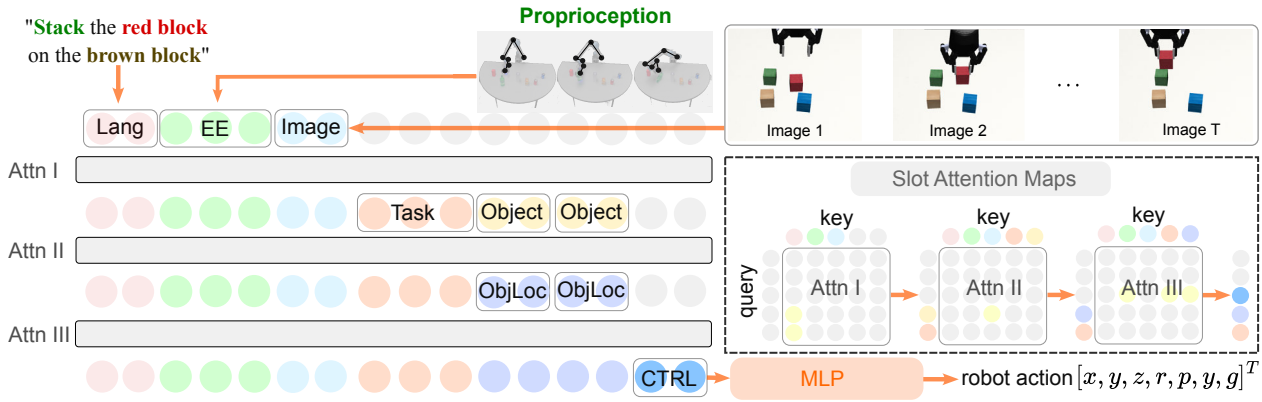

Fig. 2. LLMs decompose high-level tasks, such as "*stack the red block on the brown block*" into modules and construct a hierarchy among them. AutoMA integrates this hierarchy using slot attention mechanisms for policy learning. For instance, the `Task` and `Object` only focus on `Lang` tokens in the first attention layer, the `ObjLoc` module attends to `Image` to localizing objects, and the `CTRL` module attends to `EE`, `task`, and `ObjLoc` to generate final robot actions.

Models (VLMs) are then employed to semantically understand and label each sub-module. AutoMA implements these hierarchical modules designed by LLMs in a transformer which is modularized, through supervised attention [22]. This automated framework bridges the gap between modular and end-to-end learning, enabling the reuse of functional building blocks. In summary, our contributions are as follows: (1) **Leveraging Rich Context**: Our approach leverages rich context by decomposing tasks into hierarchical modules and semantically extracting the underlying context for each module; (2) **Improving Data Efficiency**: AutoMA enhances data efficiency by enabling the reuse of functional building blocks when transferring tasks to different embodiments; (3) **Empirical Evaluations**: AutoMA demonstrates significant improvements, achieving up to 63.3% success rate improvements in complex manipulation tasks with distractors compared to baselines.

II. Related Work

Imitation learning, particularly language-conditioned imitation learning, has achieved significant success in various manipulation tasks by exploiting task-specific features and designing targeted algorithms or collecting specialized demonstrations [6], [7], [23]–[26]. For instance, task-specific features can be embedded through discrete latent plans [12], using a learned planner to guide robot policy [27], and modifying action representations [17]. While effective, these approaches are typically not able to scale up beyond their designated tasks. Recently, there has been increased focus on developing foundational robot models that aim to assimilate extensive datasets consisting of a broad spectrum of tasks and objects [3], [4], [28]–[31]. Although these models show promising scalability and flexibility, they require a large number of human demonstrations to learn a restricted number of tasks. Alternatively, some research has focused on utilizing the grounding capabilities of foundational models within the robot manipulation pipeline, which reduces the complexity of training robot policies while still maintaining robust performance. For example, studies have applied LLMs to decompose high-level instructions into lower-level skills [32], generate embodied control signals [33], translate language instructions into robot behavior-based reward functions [34], finding out the affordance of the different parts of objects [35], or even outputting low-level control signals directly [36]. However, these methods mainly facilitate effective language grounding for robot policies, lacking the necessary visual or physical grounding of the environment. Therefore, a promising direction for scaling up involves leveraging both LLMs and Vision Language Models (VLMs). For example, the framework in [37] integrates language-embedded radiance fields to distill visuo-linguistic representations suitable for manipulation tasks. Similarly, the approach in [38] directly utilizes VLM outputs to compose 3D value maps for model-based planning. Additionally, the framework in [39] learns a waypoint-based policy conditioned on 3D keypoints by leveraging Cliport [40]. Another case is utilizing open-vocabulary object detectors in the inference, whose output is supplied as input of robot policies [41]. These methodologies enable the use of LLMs and VLMs to extract task context during inference. While this approach simplifies the process, querying large models during inference remains suboptimal. In our work, we also employ both LLMs and VLMs for enhanced data understanding and context extraction. Unlike previous methods, we distill the task and visual context into a proposed modular-based policy, eliminating the need to query large models during inference. Consequently, our approach systematically leverages the rich context within the data while mitigating computational overhead and reducing dependency on external systems.

III. AutoMA: Automated Modular Attention

Our method focuses on training a language-conditioned robot policy from a dataset $\mathcal{D} = \{\boldsymbol{d}_0, ..., \boldsymbol{d}_N\}$, which comprises N expert demonstrations. Each demonstration $\boldsymbol{d}$ is a sequence of T steps $((\boldsymbol{a}_0, \boldsymbol{s}_0, \boldsymbol{I}_0), \ldots, (\boldsymbol{a}_T, \boldsymbol{s}_T, \boldsymbol{I}_T))$. The goal is to derive a robot policy $\pi_{\boldsymbol{\theta},h}(\boldsymbol{a}|\boldsymbol{s}, \boldsymbol{I})$ that follows human instruction $\boldsymbol{s}$ based on an observed image $\boldsymbol{I}$, parameterized by $\boldsymbol{\theta}$ with a transformer deep network. The hierarchy h defines the architecture of the modules to be instantiated in the transformer in order to accomplish the task.

The AutoMA pipeline operates in two key steps: (1) **Automatic Hierarchy and Module Design.** In this phase, we employ LLMs and VLMs for task comprehension, which autonomously generates a hierarchy of modules relevant to the task, along with the corresponding training labels. (2) **End-to-End Training of Modular Attention.** Once the training data and module design are established, the second step involves training a transformer model end-to-end, embedding the modular hierarchy to guide the learning process effectively.

For instance, in a stacking task illustrated in Figure 1, the task is automatically decomposed by LLMs into several sub-tasks: understanding the required action type, e.g., (1) identifying which object to manipulate, (2) locating the object in the image, and (3) generating the end-effector controls. As shown in Figure 3, these sub-tasks are automatically formed as a hierarchical structure by LLMs, denoted as h with modules, where `Lang`, `Image`, and `EE` are considered as input modules, `Task`, `Object`, `ObjLoc`, and `CTRL` are sub-task modules. Drawing from [42], we embed this hierarchical structure h into the transformer network through modular attention, as illustrated in Figure 2. This modular design enables end-to-end training with sub-task labels. In the subsequent sections, we address (1) the automatic identification of sub-tasks and their hierarchical structure, and (2) the integration of this hierarchy into the policy network.

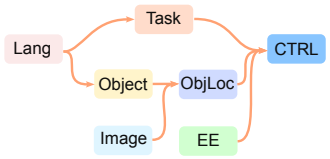

Fig. 3. Hierarchy design of a high-level task through modules.

A. Step 1: Automatic Synthesis of Hierarchy and Modules

Task Hierarchies. Task hierarchies h encapsulate the semantic structure of tasks, requiring general commonsense knowledge for generation. To achieve this, we leverage vision-enhanced Large Language Models (LLMs) due to their strong semantic interpretation capabilities. Few-shot in-context learning is employed, where the prompt template includes three key components: (a) a verbal task instruction, (b) frames from an expert demonstration, and (c) an expert-designed task hierarchy as a reference example. Our experiments show that LLMs effectively and accurately generate task hierarchies, demonstrating their potential in structuring complex tasks[1].

Sub-Task Labels. The primary challenge in identifying sub-task labels is ensuring accurate labeling for each module type while maintaining high label quality. To address this, we categorize the sub-task labels as follows:

- **Semantic Understanding**: this set of label are for `Task` and `Object` modules, where the task/action and the object types are identified at the language level.
- **Visual Grounding**: This set of labels is utilized for the `ObjLoc` module. We leverage open-vocabulary vision foundation models, such as Owl-ViT [43], for obtaining objection locations.
- **Embodiment data**: These data include end-effector position and orientation, as well as robot joints trajectory for `EE` module.

[1]For more details on how LLMs design task hierarchies, please refer to `https://auto-ma.github.io/`

B. Step 2: End-to-End Training of Modular Attention

After the task hierarchy and modules have been automatically synthesized, the next step is to embed this structure into the policy network. AutoMA achieves this by leveraging an attention mechanism to effectively guide the flow of information between the sub-modules during training.

Generally, attention mechanism operates using three components: queries (Q), keys (K), and values (V). The query (Q) identifies the most relevant keys (K), producing scores that reflect their alignment. These scores are normalized and used to weight the corresponding values (V), thereby aggregating the most relevant information.

In the proposed approach, we demonstrate that sub-modules within the hierarchy h integrate seamlessly into the attention mechanism framework. As depicted in Figure 4, the `ObjLoc` module queries both the `Image` and `Object` modules. The attention layers identify the most relevant keys—specifically, the image patch containing the object—and retrieve the corresponding values as output. Similarly, in the `Object` module, the relevant keys and values are language inputs, where the query targets language tokens, which are then fetched and used as output. This approach allows us to use the attention mechanism flexibly, fitting different tasks within the same framework. The full hierarchy h in a single transformer architecture is shown in Figure 2. The training process of AutoMA contains two main parts: aligning attention flow with the hierarchy and optimizing each module's output for its specific sub-task.

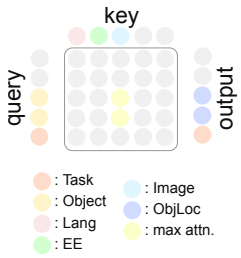

Fig. 4. The `ObjLoc` module in detail.

Training: As illustrated in Figure 2 (right), tokens between layers represent sub-modules, while the attention map shows data flow between these sub-modules. The training process of the transformer network involves two main parts: aligning attention flow with the hierarchy and optimizing each module's output for its specific sub-task. We propose the task hierarchy loss as:

$$\mathcal{L}_h = \sum_{n=0}^{N} (\text{softmax}\left(\frac{\boldsymbol{q}_n \boldsymbol{k}_n^T}{\sqrt{d}}\right) - 1)^2. \tag{1}$$

The task hierarchy loss $\mathcal{L}_h$ optimizes towards fetching the key k_n for the given module's query q_n. Therefore, it can guide the attention of the sub-modules to enforce the controlled data flow. We calculate the loss for every sub-module $1 \leq n \leq N$, $\mathcal{L}_h$. The sub-task loss $\mathcal{L}_{sub}$ is:

$$\mathcal{L}_{sub} = \sum_{n=0}^{N} \langle \text{MLP}_n(\boldsymbol{o}_n), l_n \rangle, \tag{2}$$

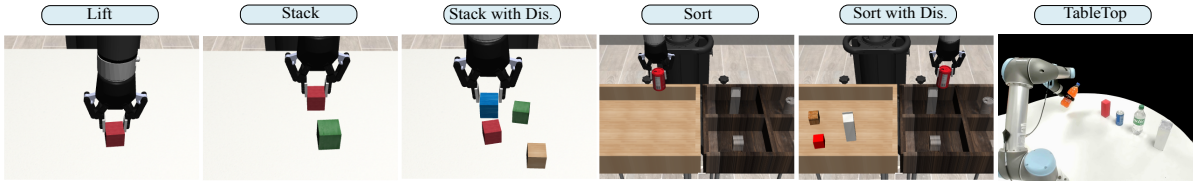

Fig. 5. Tasks (a)-(f) show examples of diverse tasks in our study, with each consisting of observation and language pairs. Tasks (a)-(e) are in simulation, while task (f) represents a real-world task.

TABLE I
TASK PROPERTIES

Task	Dis.	HiPrec	Dem.	Act.	Steps
Lift.	0	×	300	1	∼64
Stack	0	✓	300	1	∼88
Stack Dist.	3	✓	1800	1	∼97
Sort	0	✓	300	1	∼221
Sort Dist.	3	✓	1800	1	∼193
TableTop	4	×	1500	3	∼122

it the loss of every sub-modules' final output supervision. For the sub-task loss $\mathcal{L}_{sub}$, we create MLPs for each sub-module, which serve as prediction heads. The attention output token $\boldsymbol{o}_n$ of the n-th module is passed through MLP_n, which is supervised through the sub-task label l_n. The overall training objective is $\mathcal{L}_h + \mathcal{L}_{sub}$. The result of this training process is a robot policy that generates actions, which is instantiated by a transformer network and is embedded by modules that correspond for the sub-tasks.

IV. EVALUATION

We conduct a series of experiments to evaluate the efficacy of the AutoMA framework. Specifically, we aim to answer the following questions: (a) How the AutoMA performs with and without supplied rich context extracted by LLM and VLM? (b) To what extent does the AutoMA outperform the current state-of-the-art methods in terms of overall performance? (c) How can AutoMA improve data efficiency by leveraging modularity? Therefore, we evaluate the effectiveness of AutoMA across multiple manipulation tasks, each with distinct setups: (1) **task (a)-(c)** for block stacking tasks, (2) **task (d)-(e)** for object sorting tasks with and without distractors, and (3) **task (f)** for manipulation in real-world with distractors. We propose two categories of baseline policies to compare: (a) **Image-BC**: this baseline adopts an image-to-action agent framework, similar to BC-Z [6], it is built upon ResNet-18 backbone and employs FiLM [44] for conditioning using CLIP language features. (b) **Diffusion Policy** [45]: This baseline is a standard diffusion-based policy. We adopt the 1D temporal convolutional networks from [46] and construct the U-net backbone. (c) **ModAttn** [21]: This baseline is a transformer based network, which shares the same architecture with our proposed AutoMA, but trained end-to-end without enforcing rich contexts.

Task Setup: The action of the robot arm is represented as $\mathbf{a}_t$, where each action is denoted as $\mathbf{a}_t = [x, y, z, r, p, y, g]^T$, where $t \in [1, T]$. It encompasses the position of the end-effector in Cartesian coordinates (x, y, z), the orientations

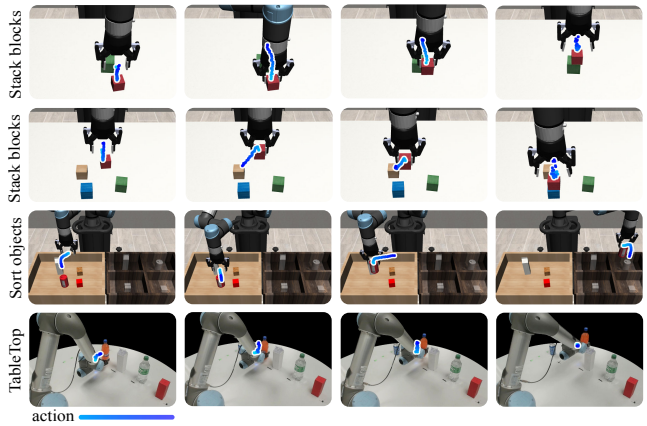

Fig. 6. The first two rows illustrate the stacking task. The third row showcases a successful sorting experiment, while the last row depicts a real-world tabletop manipulation task. The results are best appreciated with videos on the website: `https://auto-ma.github.io/`.

(r, p, y), and the gripper's joint angle g. For all the tasks, the input modalities consist of two modalities: $\mathbf{I}$ and $\mathbf{l}$. The first modality, $\mathbf{I} \in \mathbb{R}^{224\times224\times3}$, corresponds to a RGB image. The second modality, $\mathbf{l}$, refers to a language embedding derived from natural language sequences. This embedding serves as the linguistic input for the robot's understanding and decision-making processes. Table I further arranges the tasks in ascending order of subjective difficulty, providing a summary of task characteristics such as the number of distractors (Dis), number of expert demonstrations (Dem), number of varied actions (Act), and whether high-precision (HiPrec) is required or not. The simulated tasks are conducted within Robosuite [47] and we used an UR5 robot for real-world tasks.

A. Results

Stacking tasks: The stacking task evaluation results are shown in Table II. Success rates for each sub-task are based on 100 trials, where success is defined as the robot successfully stacking the block without dropping it. The primary challenge lies in assessing whether the policy can accurately connect the language to the target, particularly when multiple objects are present. In the lifting and stacking tasks (task (a)-(b)), all policy networks achieve above an 80% success rate. However, when confronted with more blocks on the table, Image-BC, Diffusion Policy, and ModAttn struggle to pick up the block and place it on the correct target based on

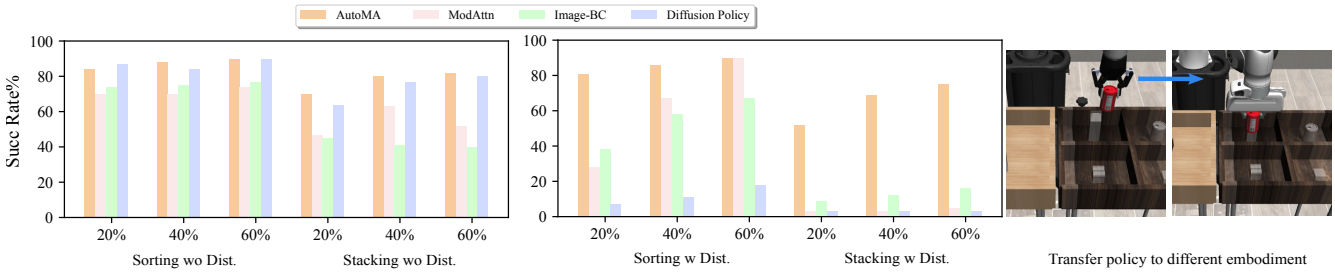

Fig. 7. Modularity can be reused when transferring the AutoMA policy from UR5 to Franka robot using different percentages of the original data, outperforming other state-of-the-art policies.

TABLE II

RESULTS EVALUATION IN FORMS OF SUCCESS RATE (%) DURING POLICY EXECUTION

Method	Stack			Sort		TableTop
	Lift	wo Dist.	w Dist.	wo Dist.	w Dist.	
Image-BC	93%	81%	25%	18%	56%	6.7%
Diffusion Policy	**100%**	81%	2%	97%	4%	6.7%
ModAttn	96%	92%	7%	88%	15%	16.7%
AutoMA	**100%**	**97%**	**81%**	**100%**	**88%**	**80%**

the language condition. From a data perspective, considering the blocks can be located at various positions with different colors, AutoMA outperforms the baselines with 81% success rate due to its ability to access informative context regarding the block locations, colors, and proprioception of the robot.

Sorting tasks: In this task, the robot picks an object based on language input and places it in the corresponding bin. The visualization of the actions are shown in Figure 6. Success rates (over 100 test trials) for sorting with and without distractors are reported in Table II. Notably, most policy networks struggle with visual and language distractors. For example, Diffusion Policy sorts a single object correctly 97% of the time but fails with varied objects (4% success rate). In contrast, AutoMA achieves a 100% success rate with one object and 88% with varied objects. This aligns with the stacking task observations, demonstrating that **AutoMA scales effectively with increased task complexity due to its hierarchical design, providing stable and contextually informed actions**.

Tabletop tasks: In this real-world tabletop manipulation task, the robot performs "pick", "push", and "rotate" actions on specified objects based on language input, such as picking up the Fanta bottle in Figure 6. According to Table II, AutoMA with rich-context achieves an 80% success rate over 30 trials. In contrast, Image-BC, Diffusion Policy, and ModAttn without rich-context achieve only 6.7%, 6.7%, and 16.7%, respectively. The most common failure is the robot cannot grasp the target object correctly due to the wrong estimation of the target position, even if it performs the correct actions. AutoMA excels at locating the target position because its object-locating module enforces high attention values on the target object. So the subsequent action generating layer is well-informed of the target position. In summary, we can conclude the proposed AutoMA framework effectively uses LLMs and VLMs to provide rich context and hierarchical design, ensuring contextually informed action generation.

Modularity Reuse: In previous sections, we demonstrate how AutoMA leverages rich context from LLMs and VLMs. Now, we address the question: "How can AutoMA improve data efficiency through modularity?". To answer this, we evaluate whether the trained modules can be transferred to another embodiment by fine-tuning with a limited amount of data as shown in Figure 7 (right). We report the success rates when transferring policies from a UR5 to a Franka robot using 20%, 40%, and 60% of the original dataset. For sorting tasks, AutoMA achieves an 81% success rate with just 20% of the new data, whereas Image-BC and Diffusion Policy achieve only 38% and 7%, respectively. Similarly, for stacking tasks, AutoMA attains a 58% success rate, while Image-BC and Diffusion Policy significantly underperform with success rates of 9% and 1%, respectively. **The reuse of trained modules allows AutoMA to transfer effectively to variations of appearances and kinematics of different embodiments with improved data efficiency**.

B. Ablation Study

Modularity Inspection: Although AutoMA is trained in an end-to-end manner, its modular design ensures full explainability of the model. A modularity inspection was conducted to assess the functionality of each sub-module within the AutoMA framework. In Table III, we report the outputs for object grounding sub-modules. Specifically, the "lang" metric indicates the accuracy of the language module in understanding the target object of the task, while the "vision" metric evaluates the success of the vision sub-modules in locating the object within the image space. The end-effector module estimates the gripper's position and orientation. We calculated the mean absolute error (MAE) between the estimated position and the ground truth. According to Table III, AutoMA is always able to understand what is the target object from language. For visual grounding of objects, AutoMA is able to detect object pixel locations with an error of $\sim$2 pixels. when handling tasks with distractors. The end-effector module demonstrates stable and accurate performance, with euclidean distance of only $\sim$0.2 cm.

Quality of Hierarchy: To further evaluate the effectiveness of the generated task hierarchy, we conducted an ablation study by comparing the resulting performance of three hierarchies, h_1, h_2, and h_3, on AutoMA (shown in Figure 9). Here, h_1 represents a "good" hierarchy with

TABLE III

EVALUATION FOR OBJECT GROUNDING AND END-EFFECTOR LOCATION.

Task	1^{st} Object Grounding		2^{nd} Object Grounding		End-Effector
	Lang	Vision (px)	Lang	Vision (px)	(cm)
Lift	100%	0.46 ± 0.34	-	-	0.23 ± 0.13
Stack wo Dist.	100%	0.45 ± 0.35	100%	1.37 ± 1.42	0.21 ± 0.12
Stack w Dist.	100%	1.49 ± 3.57	100%	1.99 ± 2.44	0.18 ± 0.09
Sort wo Dist.	100%	0.51 ± 0.35	-	-	0.17 ± 0.09
Sort w Dist.	100%	1.92 ± 1.14	-	-	0.29 ± 0.13

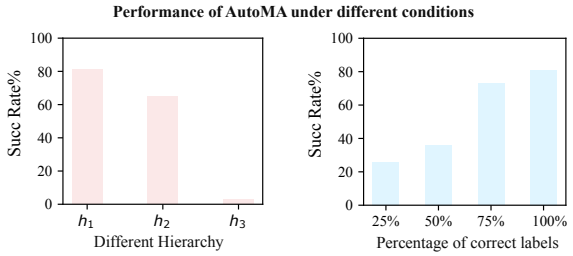

Fig. 8. AutoMA performance on "stack with distractors" task with different quality of hierarchy and VLM labels.

correct information flow, while h_2 does ignores the state of the robot end-effector, and h_3 lacks the module of finding object locations from the images. The results, presented in Figure 8 (left), demonstrate that implementing a hierarchy with incorrect modules as in h_2, leads to a 16% decrease in the success rate for the stacking with distractor task. Moreover, if there lack of a module for finding object locations, the performance drops drastically. Qualitatively, the trained policy is not able to associate language with different target objects, resulting in converging to the mean of all demonstrations.

Quality of VLM labels: In many real-world tasks, there is no guarantee that VLMs will consistently produce accurate sub-task labels. To address this, we conducted an ablation study simulating scenarios with varying degrees of missing or invalid sub-task labels to evaluate the performance of AutoMA under these conditions. As shown in Figure 8 (right), a higher number of correct labels correlated with increased success rates. Notably, when at least 75% of the labels were accurate, AutoMA maintained robust performance, indicating that VLMs do not need to produce 100% correct labels for effective task execution.

Growth of Hierarchy: The reuse of hierarchical structures facilitates not only the transfer of policies across different embodiments but also the progressive growth of these hierarchies. Previously trained modules can be effectively integrated and utilized in subsequent stages alongside newly added modules. As illustrated in Figure 9, we initially trained AutoMA on the lifting task, represented by h_{lift}, achieving a 100% success rate. We then expanded this hierarchy to h_{stack} by incorporating new vertices for recognizing the green cube and fine-tuning them accordingly, resulting in a 95% success rate. The enhanced model was subsequently applied to the stacking task with distractors, denoted as $h_{\text{stack\_dist}}$, where the red and green cube localizers were replaced by general cube localizers for any indicated colors. Our results show that the performance on the stacking task with distractors reached

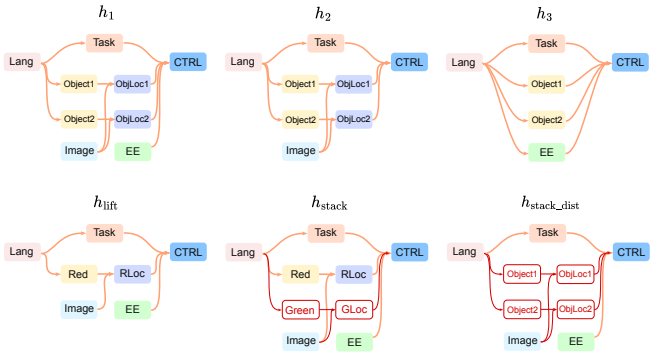

Fig. 9. The upper row shows the representations of hierarchies for the ablation study of hierarchy quality. The lower row shows the hierarchies used for h_{lift}, h_{stack}, and $h_{\text{stack\_dist}}$. The red denotes newly added modules.

an 85% success rate, surpassing the success rate achieved by training solely on task-specific data from scratch.

V. CONCLUSIONS

In this paper, we explore the following question: Can additional insights be extracted from expert demonstrations to enhance policy learning in an automatic manner? Our proposed method, AutoMA, affirmatively addresses this question. We demonstrate that leveraging Large Language Models (LLMs) and Vision Language Models (VLMs) allows AutoMA to utilize context-rich supervision signals for imitation learning. Through experimentation, we showcase that our proposed approach yields superior performance across a diverse array of complex manipulation tasks, outperforming state-of-the-art methods by up to 63.3%. Further experiments validate the reusability of learned modules when transferring tasks to new domains. For future work, we plan to explore AutoMA's capacity to scale hierarchical module designs for decomposing longer-horizon tasks. Additionally, we anticipate that with advancements in LLMs and VLMs, AutoMA will enable the completion of more sophisticated tasks.

Limitations: While the proposed AutoMA framework effectively incorporates task context, extracting relevant context from highly sophisticated tasks, such as robotic soccer or chess, presents considerable challenges. The primary issue lies in embedding complex context through task decomposition. The LLMs leveraged for context extraction and subtask supervision may falter in generating appropriate task hierarchies for such complex activities, while VLMs might occasionally fail to detect key grasping points of challenging objects, e.g., deformable objects like strings or clothes, thereby impairing the trained policy's efficacy. It's worth noting that while our method is modularized, it currently lacks a memory module to support stateful behaviors. While it's worth noting that many state-of-the-art policy networks also lack support for statefulness [6], [17], [45], [48], this limitation can restrict our method from effectively learning highly stateful behaviors, such as periodic actions or tasks in dynamically changing environments.

REFERENCES

[1] B. D. Argall, S. Chernova, M. Veloso, and B. Browning, "A survey of robot learning from demonstration," *Robotics and autonomous systems*, vol. 57, no. 5, pp. 469–483, 2009.

[2] M. Zare, P. M. Kebria, A. Khosravi, and S. Nahavandi, "A survey of imitation learning: Algorithms, recent developments, and challenges," *IEEE Transactions on Cybernetics*, 2024.

[3] A. Brohan, N. Brown, J. Carbajal, Y. Chebotar, X. Chen, K. Choromanski, T. Ding, D. Driess, A. Dubey, C. Finn, *et al.*, "Rt-2: Vision-language-action models transfer web knowledge to robotic control," *arXiv preprint arXiv:2307.15818*, 2023.

[4] A. O'Neill, A. Rehman, A. Maddukuri, A. Gupta, A. Padalkar, A. Lee, A. Pooley, A. Gupta, A. Mandlekar, A. Jain, *et al.*, "Open x-embodiment: Robotic learning datasets and rt-x models," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2024, pp. 6892–6903.

[5] C. Chi, S. Feng, Y. Du, Z. Xu, E. Cousineau, B. Burchfiel, and S. Song, "Diffusion policy: Visuomotor policy learning via action diffusion," in *Proceedings of Robotics: Science and Systems (RSS)*, 2023.

[6] E. Jang, A. Irpan, M. Khansari, D. Kappler, F. Ebert, C. Lynch, S. Levine, and C. Finn, "Bc-z: Zero-shot task generalization with robotic imitation learning," in *Conference on Robot Learning*. PMLR, 2022, pp. 991–1002.

[7] S. Stepputtis, J. Campbell, M. Phielipp, S. Lee, C. Baral, and H. Ben Amor, "Language-conditioned imitation learning for robot manipulation tasks," *Advances in Neural Information Processing Systems*, vol. 33, pp. 13 139–13 150, 2020.

[8] H. B. Amor, G. Neumann, S. Kamthe, O. Kroemer, and J. Peters, "Interaction primitives for human-robot cooperation tasks," in *2014 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2014, pp. 2831–2837.

[9] S. Ruan, W. Liu, X. Wang, X. Meng, and G. S. Chirikjian, "Primp: Probabilistically-informed motion primitives for efficient affordance learning from demonstration," *IEEE Transactions on Robotics*, 2024.

[10] J. Aldaco, T. Armstrong, R. Baruch, J. Bingham, S. Chan, K. Draper, D. Dwibedi, C. Finn, P. Florence, S. Goodrich, *et al.*, "Aloha 2: An enhanced low-cost hardware for bimanual teleoperation," *arXiv preprint arXiv:2405.02292*, 2024.

[11] S. Belkhale, Y. Cui, and D. Sadigh, "Data quality in imitation learning," *Advances in Neural Information Processing Systems*, vol. 36, 2024.

[12] O. Mees, L. Hermann, and W. Burgard, "What matters in language conditioned robotic imitation learning over unstructured data," *IEEE Robotics and Automation Letters*, vol. 7, no. 4, pp. 11 205–11 212, 2022.

[13] S. Nair, A. Rajeswaran, V. Kumar, C. Finn, and A. Gupta, "R3m: A universal visual representation for robot manipulation," in *Conference on Robot Learning*. PMLR, 2023, pp. 892–909.

[14] P.-L. Guhur, S. Chen, R. G. Pinel, M. Tapaswi, I. Laptev, and C. Schmid, "Instruction-driven history-aware policies for robotic manipulations," in *Conference on Robot Learning*. PMLR, 2023, pp. 175–187.

[15] X. Liu, F. C. Weigend, Y. Zhou, and H. B. Amor, "Enabling stateful behaviors for diffusion-based policy learning," in *ICRA 2024 Workshop Back to the Future: Robot Learning Going Probabilistic*.

[16] A. Galashov, J. S. Merel, and N. Heess, "Data augmentation for efficient learning from parametric experts," *Advances in Neural Information Processing Systems*, vol. 35, pp. 31 484–31 496, 2022.

[17] S. Belkhale, Y. Cui, and D. Sadigh, "Hydra: Hybrid robot actions for imitation learning," in *Conference on Robot Learning*. PMLR, 2023, pp. 2113–2133.

[18] S. James and A. J. Davison, "Q-attention: Enabling efficient learning for vision-based robotic manipulation," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 1612–1619, 2022.

[19] X. Liu, Y. Yoshimitsu, and H. B. Amor, "Learning soft robot dynamics using differentiable kalman filters and spatio-temporal embeddings," in *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2023, pp. 2550–2557.

[20] R. Csordás, S. van Steenkiste, and J. Schmidhuber, "Are neural nets modular? inspecting functional modularity through differentiable weight masks," in *International Conference on Learning Representations*, 2020.

[21] Y. Zhou, S. Sonawani, M. Phielipp, S. Stepputtis, and H. Amor, "Modularity through attention: Efficient training and transfer of language-conditioned policies for robot manipulation," in *Conference on Robot Learning*. PMLR, 2023, pp. 1684–1695.

[22] Y. Zhou, S. Sonawani, M. Phielipp, H. Ben Amor, and S. Stepputtis, "Learning modular language-conditioned robot policies through attention," *Autonomous Robots*, vol. 47, no. 8, pp. 1013–1033, 2023.

[23] A. Singh, R. Hu, V. Goswami, G. Couairon, W. Galuba, M. Rohrbach, and D. Kiela, "Flava: A foundational language and vision alignment model," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 15 638–15 650.

[24] M. Shridhar, L. Manuelli, and D. Fox, "Perceiver-actor: A multi-task transformer for robotic manipulation," in *Conference on Robot Learning*. PMLR, 2023, pp. 785–799.

[25] L. X. Shi, A. Sharma, T. Z. Zhao, and C. Finn, "Waypoint-based imitation learning for robotic manipulation," in *Conference on Robot Learning*. PMLR, 2023, pp. 2195–2209.

[26] Y. Zhu, Z. Jiang, P. Stone, and Y. Zhu, "Learning generalizable manipulation policies with object-centric 3d representations," in *7th Annual Conference on Robot Learning*, 2023.

[27] C. Wang, L. Fan, J. Sun, R. Zhang, L. Fei-Fei, D. Xu, Y. Zhu, and A. Anandkumar, "Mimicplay: Long-horizon imitation learning by watching human play," in *7th Annual Conference on Robot Learning*, 2023.

[28] A. Brohan, N. Brown, J. Carbajal, Y. Chebotar, J. Dabis, C. Finn, K. Gopalakrishnan, K. Hausman, A. Herzog, J. Hsu, *et al.*, "Rt-1: Robotics transformer for real-world control at scale," *arXiv preprint arXiv:2212.06817*, 2022.

[29] H. R. Walke, K. Black, T. Z. Zhao, Q. Vuong, C. Zheng, P. Hansen-Estruch, A. W. He, V. Myers, M. J. Kim, M. Du, *et al.*, "Bridgedata v2: A dataset for robot learning at scale," in *Conference on Robot Learning*. PMLR, 2023, pp. 1723–1736.

[30] Octo Model Team, D. Ghosh, H. Walke, K. Pertsch, K. Black, O. Mees, S. Dasari, J. Hejna, C. Xu, J. Luo, T. Kreiman, Y. Tan, L. Y. Chen, P. Sanketi, Q. Vuong, T. Xiao, D. Sadigh, C. Finn, and S. Levine, "Octo: An open-source generalist robot policy," in *Proceedings of Robotics: Science and Systems*, Delft, Netherlands, 2024.

[31] S. Belkhale, T. Ding, T. Xiao, P. Sermanet, Q. Vuong, J. Tompson, Y. Chebotar, D. Dwibedi, and D. Sadigh, "Rt-h: Action hierarchies using language," *arXiv preprint arXiv:2403.01823*, 2024.

[32] A. Brohan, Y. Chebotar, C. Finn, K. Hausman, A. Herzog, D. Ho, J. Ibarz, A. Irpan, E. Jang, R. Julian, *et al.*, "Do as i can, not as i say: Grounding language in robotic affordances," in *Conference on robot learning*. PMLR, 2023, pp. 287–318.

[33] J. Liang, W. Huang, F. Xia, P. Xu, K. Hausman, B. Ichter, P. Florence, and A. Zeng, "Code as policies: Language model programs for embodied control," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2023, pp. 9493–9500.

[34] W. Yu, N. Gileadi, C. Fu, S. Kirmani, K.-H. Lee, M. G. Arenas, H.-T. L. Chiang, T. Erez, L. Hasenclever, J. Humplik, *et al.*, "Language to rewards for robotic skill synthesis," in *7th Annual Conference on Robot Learning*, 2023.

[35] S. Li, S. Bhagat, J. Campbell, Y. Xie, W. Kim, K. Sycara, and S. Stepputtis, "Shapegrasp: Zero-shot task-oriented grasping with large language models through geometric decomposition," *arXiv preprint arXiv:2403.18062*, 2024.

[36] S. Mirchandani, F. Xia, P. Florence, B. Ichter, D. Driess, M. G. Arenas, K. Rao, D. Sadigh, and A. Zeng, "Large language models as general pattern machines," in *Conference on Robot Learning*. PMLR, 2023, pp. 2498–2518.

[37] A. Rashid, S. Sharma, C. M. Kim, J. Kerr, L. Y. Chen, A. Kanazawa, and K. Goldberg, "Language embedded radiance fields for zero-shot task-oriented grasping," in *Conference on Robot Learning*. PMLR, 2023, pp. 178–200.

[38] W. Huang, C. Wang, R. Zhang, Y. Li, J. Wu, and L. Fei-Fei, "Voxposer: Composable 3d value maps for robotic manipulation with language models," in *Conference on Robot Learning*. PMLR, 2023, pp. 540–562.

[39] P. Sundaresan, S. Belkhale, D. Sadigh, and J. Bohg, "Kite: Keypoint-conditioned policies for semantic manipulation," in *Conference on Robot Learning*. PMLR, 2023, pp. 1006–1021.

[40] M. Shridhar, L. Manuelli, and D. Fox, "Cliport: What and where pathways for robotic manipulation," in *Conference on robot learning*. PMLR, 2022, pp. 894–906.

[41] A. Stone, T. Xiao, Y. Lu, K. Gopalakrishnan, K.-H. Lee, Q. Vuong, P. Wohlhart, S. Kirmani, B. Zitkovich, F. Xia, *et al.*, "Open-world

object manipulation using pre-trained vision-language models," in *Conference on Robot Learning*. PMLR, 2023, pp. 3397–3417.

[42] F. Locatello, D. Weissenborn, T. Unterthiner, A. Mahendran, G. Heigold, J. Uszkoreit, A. Dosovitskiy, and T. Kipf, "Object-centric learning with slot attention," *Advances in neural information processing systems*, vol. 33, pp. 11 525–11 538, 2020.

[43] M. Minderer, A. Gritsenko, A. Stone, M. Neumann, D. Weissenborn, A. Dosovitskiy, A. Mahendran, A. Arnab, M. Dehghani, Z. Shen, *et al.*, "Simple open-vocabulary object detection," in *European Conference on Computer Vision*. Springer, 2022, pp. 728–755.

[44] E. Perez, F. Strub, H. De Vries, V. Dumoulin, and A. Courville, "Film: Visual reasoning with a general conditioning layer," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 32, 2018.

[45] C. Chi, S. Feng, Y. Du, Z. Xu, E. Cousineau, B. Burchfiel, and S. Song, "Diffusion policy: Visuomotor policy learning via action diffusion," in *Proceedings of Robotics: Science and Systems (RSS)*, 2023.

[46] M. Janner, Y. Du, J. Tenenbaum, and S. Levine, "Planning with diffusion for flexible behavior synthesis," in *International Conference on Machine Learning*, 2022.

[47] Y. Zhu, J. Wong, A. Mandlekar, R. Martín-Martín, A. Joshi, S. Nasiriany, and Y. Zhu, "robosuite: A modular simulation framework and benchmark for robot learning," *arXiv preprint arXiv:2009.12293*, 2020.

[48] T. Z. Zhao, V. Kumar, S. Levine, and C. Finn, "Learning Fine-Grained Bimanual Manipulation with Low-Cost Hardware," in *Proceedings of Robotics: Science and Systems*, July 2023.