Adaptive Visual Abstraction via Object Token Merging and Pruning for Efficient Robot Manipulation

Jisu Han
Graduate School of AI at KAIST
jshan@kaist.ac.kr

Abstract

Robots must efficiently manipulate objects in complex, unstructured environments. This entails identifying task-relevant objects, which consist of objects that are directly connected to the goal and constraint objects that may cause collisions during robot execution. Leveraging foundation models like Vision-Language Model or CLIP holds promise, yet they usually lack awareness of the robot's configuration and fail to recognize constraint objects, resulting in sub-optimal performance. Fine-grained object segments offer an alternative but are computationally expensive. Humans instinctively process information about objects in a manner that aligns with the demands of the task and trajectory requirements. Inspired by this, we propose integrating an architectural bias into imitation learning framework. By merging and pruning object tokens based on task relevance and importance, our method, named as ***GoS****, reduces computational burdens and enhances task understanding, leading to higher success rates. Applied to vision-based multi-task articulated object manipulation domain, our approach shows 1.7× higher success rate in general scenes, 1.6 × higher success rate in scenes where constraint objects exist, and 3× less computation cost.*

1. Introduction

Consider a scenario where a robot needs to open an oven door, surrounded by various objects. In environments cluttered with numerous objects, the robot must identify objects relevant to its task, while ignoring irrelevant distractors. Essential objects for this task could include the oven knob or the robot's arm, which are directly linked to the goal. Additionally, the robot might need to consider other objects that could influence its trajectory from the arm to the knob, such as a notebook or information about the oven. We name these as constraint objects, and without awareness of them, the robot is likely to collide during its movement, eventually resulting in sub-optimal performance [18]. Our goal is for the robot to effectively manipulate objects in such scenes where a huge number of objects, including constraint objects and distractor objects, exist. This is challenging because it requires thorough object reasoning to identify objects that are relevant to the goal, robot execution, and the environment.

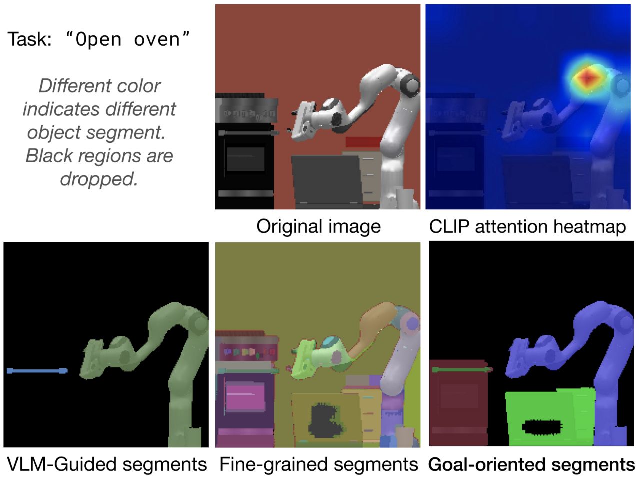


Figure 1. Given the task description and RGB-D image, the robotic agent should determine the next action. For successful task execution, it is essential for the robot to recognize objects relevant to the task, including goal-related objects and constraint objects that could potentially cause collisions.

While leveraging foundation models such as Vision-Language Model (VLM) [12, 16, 22] or CLIP [8, 15, 19] is a promising direction, the lack of awareness about the robot's current configuration often overlook constraint objects. As illustrated in VLM-guided segment of Figure 1, constraint objects, such as the surrounding objects of the robot, are not identified. To circumvent this, one alternative solution is fetching fine-grained object segments. However, storing and processing them as object-centric representations using transformer-based models incurs significant computational cost, especially due to the involvement of multi-head self-attention mechanisms, making real-time operation challenging.

Humans naturally process object information at a level suited to task and trajectory requirements, filtering out unnecessary details and grouping segments of relevant objects based on the requirements of the task [6]. As shown in the goal-oriented segments of Figure 1, objects that are not essential to the goal, like the oven button, are filtered out, while crucial objects are identified. The robot and its associated segments are grouped into a single object token. Similarly, constraint objects are also consolidated into a unified token of information relevant to the task. Inspired by this, we propose integrating an architectural bias in imitation learning framework. We utilize the token merging and pruning technique from Vision Transformer [7] literature into the policy network; after converting fine-grained object segments into object tokens, we merge these tokens based on task relevance and prune them according to task importance. The remaining object tokens may include constraints, robot information, or task-related object. This process not only reduces computational burdens but also enhances task understanding by identifying crucial objects through the pruning process, leading to higher success rates.

We apply this intuition to vision-based multi-task articulated object manipulation domain. Injecting such bias shows 1.7× higher success rate in general scenes, 1.6 × higher success rate in scenes where constraint objects exist, and 3× less computation cost. In summary, the main contributions of this paper are twofold:

1. We propose a method that incorporates visual abstraction into object-centric representations by selectively merging and pruning object information based on its relevance and importance to the task.
2. We empirically show that our method is significantly more efficient in terms of performance and computational cost than others that do not utilize this architecture and models that utilize foundation models, even when scenes where constraint objects exist.

2. Related works

Vision-based Robot Manipulation. Recent papers have processed the given observation image by utilizing foundation models, specifically fetching goal objects [2, 18, 19]. Through VLMs, we may selectively extract a list of objects that are directly associated with the goal. However, there's a high risk of task failure due to overlooking constraint-related objects that could interfere with robot trajectory execution. To cover those constraints, we may employ a universal instance segmentation model, such as Segment anything (SAM) [12], to access and use the entire image segments. To utilize this, since the segment level required by relevant objects varies depending on the task, ultimately, extracting segments at the most fine-grained level will be necessary. However, extracting and computing all of these segments incur high computational costs. In this study, we propose goal-oriented segmentation that ensures computational efficiency and robust handling of distractors by forming the necessary segments tailored to the task.

Token Merging and Pruning. Vision transformer (ViT) [7] suffers from the huge computing cost, which is due to the quadratic time complexity of multi-head self-attention. Recent papers [3, 4, 13, 14, 17, 21] have attempted to solve this issue by pruning down tokens to remove irrelevant information or merging tokens to reduce the number of tokens within each Transformer block. Taking inspiration from this token merging and pruning process, our approach is built on the architecture to merge or prune different object tokens.

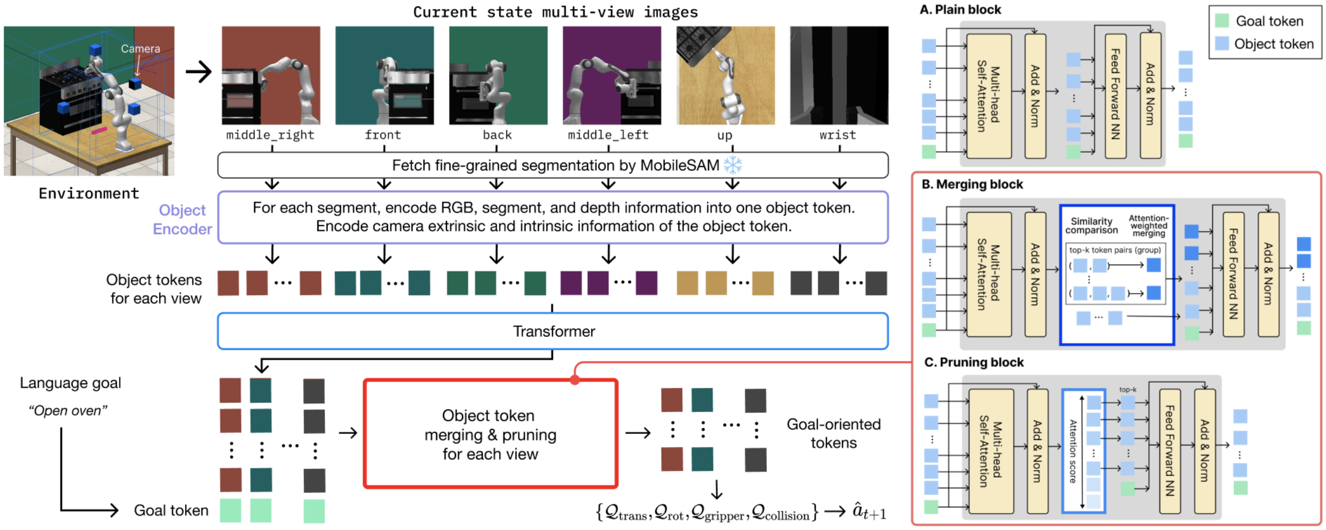
3. Methods

Our goal is to learn an imitation learning policy network that learns how to group and prune task-relevant visual elements in an end-to-end manner, illustrated in Figure 2. Given a language description of the task and the current multi-view RGB-D images from cameras, the policy predicts the robot's next action, specified by a target end-effector pose and gripper state as in [11].

To consider the relevance of the task and the objects, we design the network in an object-centric manner. To be specific, all object segments are fetched by foundation model [12, 22] at a fine-grained level. Each object's segment, depth, and RGB information are encoded into a single object token by Vision Transformer [7].

Our model is designed under transformer encoder architecture to take various numbers of object tokens as input. First, we process across different object tokens in different views. Objects for each view are processed in a transformer encoder model where merging and blocking blocks are embedded. Compared to the plain transformer block((b) block A in Figure 2), on our network, merging ((b) block B in Figure 2) and pruning ((b) block C in Figure 2) blocks alternate. As each block passes, the number of object tokens is greatly reduced. In our setting, the model handles a maximum of 20 tokens, which are progressively reduced through successive blocks to 4 tokens. This reduces the computational cost, without hindering the overall performance. The following paragraphs describe each block.

In the merging block, object tokens that share the same information based on the goal are grouped into one token. For example, in the task of placing a bottle on a table, various segments of the bottle are merged into one segment for grasping. This process simplifies the robot's perception of the object by combining its segments into a cohesive token. Merging occurs after multi-head self-attention, where attention values between the goal token and each object token are computed, conditioning each object token under the goal token. Object tokens that share the same information based on the goal receive high similarity scores, therefore merged through attention value-based weighted sum. This


Current state multi-view images
Camera
Environment
middle_right
front
back
middle_left
up
wrist
Fetch fine-grained segmentation by MobileSAM
Object Encoder
For each segment, encode RGB, segment, and depth information into one object token.
Encode camera extrinsic and intrinsic information of the object token.
Object tokens for each view
Transformer
Language goal
"Open oven"
Goal token
Object token merging & pruning for each view
Goal-oriented tokens
$\{\mathcal{Q}_{\text{trans}}, \mathcal{Q}_{\text{rot}}, \mathcal{Q}_{\text{gripper}}, \mathcal{Q}_{\text{collision}}\} \rightarrow \hat{a}_{t+1}$
A. Plain block
Goal token
Object token
Multi-head Self-Attention
Add & Norm
Feed Forward NN
Add & Norm
B. Merging block
Similarity comparison
Attention-weighted merging
top-k token pairs (group)
C. Pruning block
Attention score
top-k

(a) Model architecture

(b) Original, Merging and Pruning block

Figure 2. **Overall model architecture.** (a) shows the overall model architecture. After passing the object encoder, the object tokens are passed in the transformer encoder across different views. Note that each square indicates a single object token, which originated from the object segment, and the same color indicates object tokens that are from the same camera view. Then, the object tokens are passed into the transformer encoder with token merging and pruning blocks attached. Robot's next action is determined based on the goal-oriented tokens for each view. Token merging and pruning blocks are illustrated in (b). Object tokens are progressively merged and pruned based on the similarity score across object tokens and attention score across the goal token.

ensures that the merged token represents the combined attributes of the grouped objects' information.

In the pruning block, object tokens that are irrelevant to the goal are removed. Attention scores between the goal token and the object tokens are the criteria for the object to be removed. By identifying the top-k tokens with the lowest relevance scores, the tokens that do not contribute meaningfully to the task are discarded.

The model outputs the next step action, which consists of a 7-dimensional action, including the 6-DoF target end effector pose (3-DoF for translation and 3-DoF for rotation), and 1-DoF gripper state(open or close). Initially, it predicts a heat map for each view using per-image features. These heat maps, derived from multi-views, are back-projected to calculate scores for a discretized 3D point set that covers the robot workspace. Finally, the translation of the end effector is determined by identifying the 3D point with the highest score.

We employ a combination of loss functions to train the next action in an end-to-end manner as in [9], specifically applying cross-entropy loss to heat-maps generated for each image, and for each Euler angle and the gripper state, cross-entropy loss is applied in the discretized rotation aspects.

4. Experiments

We investigate the following questions: (1) Does our model generalize better compared to methods that filter objects through a foundation model or methods that leverage all objects, particularly in tasks involving constraint objects? (2) Is our model more computationally efficient compared to methods that leverage all objects? (3) Does our processing approach, which involves both merging and pruning, lead to higher performance than using only one of these techniques?

4.1. Experiment setup

Experiments are conducted in the multi-task articulated object manipulation domain, utilizing the Franka Panda robot with a parallel-jaw gripper to complete manipulation tasks involving containers such as fridges or ovens [10]. To evaluate the robustness of the learned policies, we tested in two task variants: `Canonical` task, where the orientation and direction of articulated objects are randomly sampled, and `Challenging` task, which includes 3-4 unseen distractors and 1-2 constraint objects exist under `canonical` task setting. We evaluated each model with two metrics: the success rate (%) and the computational cost measured in GFlops.

We compare our model, named as GoS, to baselines that takes additional information from foundation model: CLIP takes CLIP attention map between task description and image, as in [8]; VLM takes task-related object segments by utilizing VLM [1, 16] as in [19, 20]; SAM takes fine-grained object segments by universal models for seg-

Model	Additional input	Model	Canonical		Challenging	Computation cost	
info		architecture	Success(%)↑	Rank↓	Success(%)↑	GFlops↓	DivMin↓
(1) Default	-	Original	16.6	5.4	11.7	31.4	1
(2) Clip	CLIP attention map	Original	19.2	5	12.5	31.4	1
(3) Vlm	VLM-guided segments	Original	24.2	4	13.3	66.0	2.1
(4) Sam	Fine-grained segments	Original	19.6	4.8	17.3	124	3.9
(5) Ours-M	Fine-grained segments	Merging	32.5	3	27.7	41.4	1.3
(6) Ours-P	Fine-grained segments	Pruning	37.2	2.6	22.2	41.3	1.3
(7) Ours-GoS	Fine-grained segments	Both	**41.1**	**1.4**	**28.5**	41.4	1.3

Table 1. **Quantitative results of multi-task articulated object manipulation domain.** We report the average success rate of all tasks, and the average rank for each task, both in `canonical` and `challenging` tasks. The values are computed over 20-30 initializations with repeated runs of 3 random seeds. Note that the DivMin column represents the GFlops divided by the minimum GFlops value.

mentation [12, 22]. Additionally, we compare to Default, which takes no additional information. Next, we compare our model with different block choices: Ours-M, which only takes merging blocks, and Ours-P, which only takes pruning blocks. For further details regarding the baselines and domain, see appendix, Section 6.2.1.

4.2. Results

We present the quantitative results in Table 1. GoS outperforms both Clip and Vlm in terms average success rate. We believe this is due to GoS's ability to process fine-grained segments, enabling a more effective consideration of constraint objects that are often overlooked by CLIP attention maps or VLM-guided segments. This argument is observed in Figure 3. Specifically, Vlm fails to recognize constraint objects, such as the drawer in the first example or the notebook in the second example. In contrast, GoS exhibits an awareness of such details.

GoS achieves higher average success rate than the Sam. We believe this is because merely processing fine-grained segments lets the model focus on fine-grained details, which makes the model more susceptible to noise. In contrast, GoS's architectural bias leads the policy to identify which segments are necessary for each specific task. This facilitates a task-oriented visual abstraction strategy within the model, enhancing task understanding and allowing for more precise and effective robotic actions.

Additionally, we observed that Vlm outperforms Sam in `canonical` tasks, yet its success rate deteriorates in `challenging` tasks. This supports the limitation in VLM's ability to reason about constraint objects.

GoS exhibits significantly lower computational costs compared to both Vlm and Sam. In Sam, the segments increase in number, which greatly escalates the computational costs due to the quadratic time complexity of multi-head self-attention. Reducing the number of tokens through token merging and pruning confirms the decrease in computational costs.

We assess the effectiveness of integrating both merging and pruning blocks by comparing ours with ours-M and ours-P. Both models show lower success rate than GoS. Specifically, ours-P shows a huge decrease in performance in `challenging` tasks. This decline is attributed to its failure to preserve essential object information, such as constraint objects.

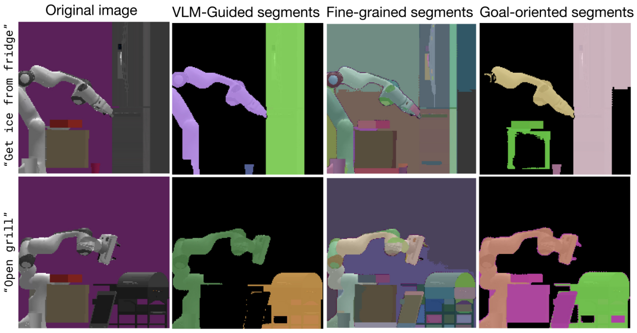


Figure 3. **Qualitative results in `Challenging` task.** We show two task example's final object token visualization from Vlm, Sam, and GoS.

5. Conclusion

In this study, we consider the problem of developing task-relevant visual abstraction in the scope of vision-based imitation learning among robot manipulation tasks. Leveraging foundation models to induce goal-related objects introduces two significant hurdles: the need for reasoning aligned with robot constraints and the substantial computational burden associated with processing find-grained segments. By utilizing token merging and pruning technique, our approach shows higher success rates and less computation cost. Through these findings, our research contributes to advancing the capabilities of robotic systems, enabling them to navigate and manipulate objects with greater precision and efficiency in real-world scenarios.

References

[1] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023. 3

[2] Shogo Akiyama, Dan Ogawa Lillrank, and Kai Arulkumaran. Fine-grained object detection and manipulation with segmentation-conditioned perceiver-actor. In *ICRA2023 Workshop on Pretraining for Robotics (PT4R)*, 2023. 2

[3] Zhe Bian, Zhe Wang, Wenqiang Han, and Kangping Wang. Muti-scale and token mergence: Make your vit more efficient. *arXiv preprint arXiv:2306.04897*, 2023. 2

[4] Daniel Bolya, Cheng-Yang Fu, Xiaoliang Dai, Peizhao Zhang, Christoph Feichtenhofer, and Judy Hoffman. Token merging: Your vit but faster. *arXiv preprint arXiv:2210.09461*, 2022. 2

[5] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020. 1

[6] Bergen R Bugelski and Delia A Alampay. The role of frequency in developing perceptual sets. *Canadian Journal of Psychology/Revue canadienne de psychologie*, 15(4):205, 1961. 2

[7] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020. 2

[8] Ainaz Eftekhar, Kuo-Hao Zeng, Jiafei Duan, Ali Farhadi, Ani Kembhavi, and Ranjay Krishna. Selective visual representations improve convergence and generalization for embodied ai. *arXiv preprint arXiv:2311.04193*, 2023. 1, 3

[9] Ankit Goyal, Jie Xu, Yijie Guo, Valts Blukis, Yu-Wei Chao, and Dieter Fox. Rvt: Robotic view transformer for 3d object manipulation. In *Conference on Robot Learning*, pages 694–710. PMLR, 2023. 3

[10] Stephen James, Zicong Ma, David Rovick Arrojo, and Andrew J Davison. Rlbench: The robot learning benchmark & learning environment. *IEEE Robotics and Automation Letters*, 5(2):3019–3026, 2020. 3, 1

[11] Edward Johns. Coarse-to-fine imitation learning: Robot manipulation from a single demonstration. In *2021 IEEE international conference on robotics and automation (ICRA)*, pages 4613–4619. IEEE, 2021. 2

[12] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C Berg, Wan-Yen Lo, et al. Segment anything. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4015–4026, 2023. 1, 2, 4

[13] Youwei Liang, Chongjian Ge, Zhan Tong, Yibing Song, Jue Wang, and Pengtao Xie. Not all patches are what you need: Expediting vision transformers via token reorganizations. *arXiv preprint arXiv:2202.07800*, 2022. 2

[14] Sifan Long, Zhen Zhao, Jimin Pi, Shengsheng Wang, and Jingdong Wang. Beyond attentive tokens: Incorporating token importance and diversity for efficient vision transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10334–10343, 2023. 2

[15] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR, 2021. 1

[16] Tianhe Ren, Shilong Liu, Ailing Zeng, Jing Lin, Kunchang Li, He Cao, Jiayu Chen, Xinyu Huang, Yukang Chen, Feng Yan, et al. Grounded sam: Assembling open-world models for diverse visual tasks. *arXiv preprint arXiv:2401.14159*, 2024. 1, 3

[17] Michael S Ryoo, AJ Piergiovanni, Anurag Arnab, Mostafa Dehghani, and Anelia Angelova. Tokenlearner: What can 8 learned tokens do for images and videos? *arXiv preprint arXiv:2106.11297*, 2021. 2

[18] Chan Hee Song, Jihyung Kil, Tai-Yu Pan, Brian M Sadler, Wei-Lun Chao, and Yu Su. One step at a time: Long-horizon vision-and-language navigation with milestones. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15482–15491, 2022. 1, 2

[19] Austin Stone, Ted Xiao, Yao Lu, Keerthana Gopalakrishnan, Kuang-Huei Lee, Quan Vuong, Paul Wohlhart, Sean Kirmani, Brianna Zitkovich, Fei Xia, et al. Open-world object manipulation using pre-trained vision-language models. *arXiv preprint arXiv:2303.00905*, 2023. 1, 2, 3

[20] Austin Stone, Ted Xiao, Yao Lu, Keerthana Gopalakrishnan, Kuang-Huei Lee, Quan Vuong, Paul Wohlhart, Sean Kirmani, Brianna Zitkovich, Fei Xia, et al. Open-world object manipulation using pre-trained vision-language models. *arXiv preprint arXiv:2303.00905*, 2023. 3

[21] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. In *International conference on machine learning*, pages 10347–10357. PMLR, 2021. 2

[22] Chaoning Zhang, Dongshen Han, Yu Qiao, Jung Uk Kim, Sung-Ho Bae, Seungkyu Lee, and Choong Seon Hong. Faster segment anything: Towards lightweight sam for mobile applications. *arXiv preprint arXiv:2306.14289*, 2023. 1, 2, 4

Adaptive Visual Abstraction via Object Token Merging and Pruning for Efficient Robot Manipulation

Supplementary Material

6. Appendix

6.1. Implementation details of our model

- Transformer architecture
  - Feature dimension: 256
  - Number of head: 6
- Training details
  - Training epoch: 90
  - # Train dataset demonstration: 100
  - # Evaluation dataset demonstration: 20
  - # Test dataset demonstration: 30 for `canonical` task, 20 for `challenging` task
- Configuration of Merging/Pruning blocks: Each model starts from maximum 20 object tokens, by passing each block (which is indicated by →), ends with few number of object tokens. We show the number of left tokens with the number of pruned, merged object tokens inside the bracket.
  1. Merging: 20 → 10 (10 merged) → 8 (2 merged) → 6 (2 merged) → 5 (1 merged) → 5 → 4 (1 merged) → 4
  2. Pruning: 20 → 10 (10 pruned) → 8 (2 pruned) → 7 (2 pruned) → 6 (1 pruned) → 6 → 4 (1 pruned) → 4
  3. Ours: 20 → 10 (10 merged) → 8 (2 pruned) → 6 (2 merged) → 5 (1 pruned) → 5 → 4 (1 merged) → 4

6.2. Experiments details

6.2.1 Baseline details

In this section, we describe the additional implementation details on producing the results of the baselines.

- CLIP: Fetch 7 × 7 attention map from CLIP, resize it into the original RGB image.
- VLM: List of object segment names are generated by GPT-3 [5], and then fetch those object segments by Vision-language model [16]. Following is the example prompt to fetch the fine-grained segments: `I am a Franka Panda robot with a parallel jaw-gripper. There is an oven in front of me, and I have to perform the task { Language goal }. Come up with a list of object segment names that are needed to execute the task successfully.`
- SAM: Object segments are fetched by foundation model for segmentation [22]. Note that the segmentation level is set to become fine-grained, and such segments are used for the following models: OURS-M, OURS-P, and OURS-GOS.

6.2.2 Multi-task details

In this section, we descibe the description of each of the task [10], along with the execution examples in Figure 4.

- `Open oven`: Grab hold of the handle and pull the oven door open
- `Turn oven on`: Grip the leftmost knob on the oben and rotate it anti-clockwise to turn the oven on
- `Open grill`: Grasp the handle and raise the cover up to open the frill
- `Close grill`: Grasp the handle and lower the grill cover to close it
- `Get ice from fridge`: Push the cup up against the tongue of the ice dispenser to retrieve ice from the fridge

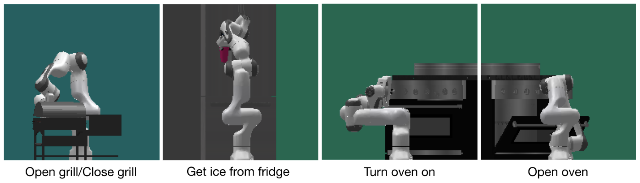
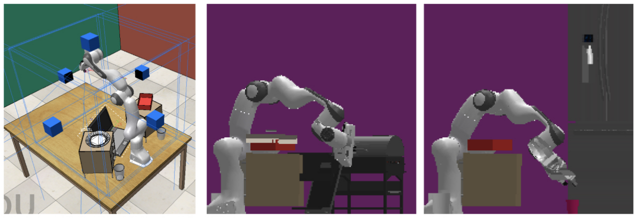


Figure 4. **Multi-task execution examples.**

To train the network, a dataset $D = \{D_1, D_2, \cdots, D_n\}$ of n expert demonstrations for each task. Each demonstration D_i is a successful roll-out, which consists of the language description, a sequence of the observations from RGBD cameras, and the sequence of corresponding robot actions. Articulated object's position and rotation are randomly sampled in each demonstration. The observations are from six RGB-D cameras positioned at the front, back, middle left, middle right, up, and wrist with a resolution of 192 × 192. Environment examples of `challenging` task in Figure 5.



(a) `Challenging` task scene

(b) `Challenging` task execution

Figure 5. `Challenging` task examples. (a) shows some distractors and constraint objects, such as notebook or drawers, that can let the robot choose alternative trajectory. (b) shows some success trajectories where those distractors are considered.