Supplementary Material
Modularity through Attention: Efficient Training and Transfer of Language-Conditioned Policies for Robot Manipulation

1 Additional Experiments

In this section, we investigate generalization capabilities along different dimensions. Experiments address the robustness to a.) to unseen colors, b.) object scaling, c.) object synonyms, d.) image occlusions, and e.) unseen object types. Summarizing the generated insights and success rates (SR) on the pushing task, we notice that our method:

- is on par or outperforms BC-Z in presence of occlusions (SR: **81.25%** at 20% occlusion)
- generalizes well to unseen object colors and color names (SR: **74.3%**)
- can deal with synonyms of object names (SR: **82.5%**)
- moderately addresses changes in object size (SR: **57.89%**)
- does not generalize well to objects not contained in the training set (SR: **34.6%**)
- can easily be extended with new modules, e.g., obstacle avoidance (SR: **88.0%**)

Below, we present technical details of above experiments. In all of the experiments, six objects were placed on the table. In addition, we demonstrate an easy and intuitive way to add new modules to an existing hierarchy by performing experiments of plugging in a new module for obstacle detection and avoidance in Sec. 1.6.

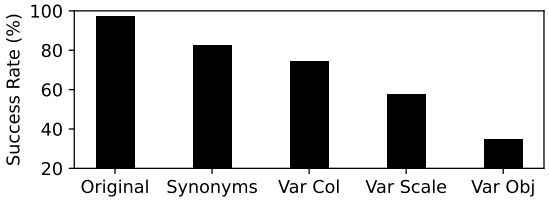

Figure 1: Success rate in generalization tasks.

1.1 Task Execution under Occlusion

In this experiment, we evaluated the ability of our previously trained model to deal with occlusions in the input image. To this end, we black out an image patch from the camera feed, thereby partially covering the target object. We performed experiments with varied mask sizes of 4, 6, 8, 10 pixels, which result in covering approximately 20%, 42%, 68% and 80% of the object area. To accurately measure the coverage we calculate the number of pixels of the object that are affected (occluded) by the mask. We also performed the same experiment with the best BC-Z model generated in our previous experiments. Fig. 2 depicts the results of this experiment. Note that at test time, six objects are placed on the table. All of 3 tasks (pick, putdown and push) are incorperated in this experiment. For small masks and occlusion rates of up to 20%, the success rate of our model is only marginally affected with a drop of about 1.1%. However, the BC-Z model saw a drop of about 9.35%. For occlusion rates of 40% or more, no significant differences between our model and BC-Z can be noticed. both our model and BC-Z are comparable in performance. In summary, our modular method shows resilience to occlusions that is at least on par, if not better, than BC-Z.

1.2 Task Execution with Unseen Colors

6th Conference on Robot Learning (CoRL 2022), Auckland, New Zealand.

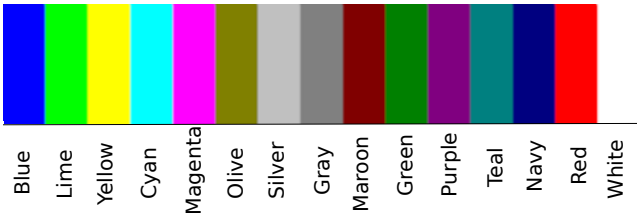

Figure 3: Color used for the generalization test.

In this experiment, we investigated the ability of our model to identify objects when colors are changed. As reported before, the model was trained with 6 objects whose colors were: red, green, blue, white and brown. Hence, five colors were present in the original training set. For this experiment, we changed the colors of the objects by randomly assigning them a color from the palette in Fig. 3.

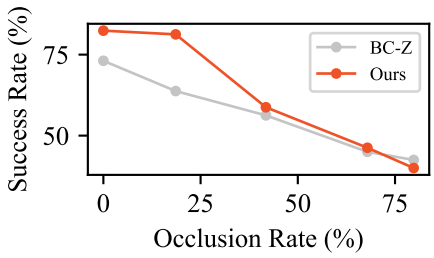

Figure 2: Success rate when part of the target object is occluded.

In each test run, six objects are placed on the table and the target object is referred to by color in the instruction, i.e., "Push the navy object!". Object names are not used in order to ensure that the model correctly identifies the color in both the verbal instruction and the visual image feed. At test time, six random objects are placed on the table. The achieved success rate is 74.3% for the push task. A possible explanation of this high success rate is that the primary colors were part of the training set and that the training successfully aligned the embedding of the visual inputs and the linguistic inputs (image features vs. language features) to enable generalization to new colors. Going forward, we will analyse this result in more detail to better understand the process by which this generalization came to be. A crucial aspect of this experiment is that the model does not only choose the next best color by mapping a novel color to a learned one. Instead, the model is able to accurately distinguish the novel colors in the presence of all learned colors, showing true generalization to novel aspects.

1.3 Task Execution with Scaled Objects

In this experiment, we investigated the resilience of the trained model to changes in object size. We apply a scaling factor of between 0.5x and 3x to scene objects. We first randomly sample the number of dimensions that will be modified and, subsequently, sample scaling factors in the above range. Hence, the number of dimensions being modified is variable. Again, at test time six objects are on the table, with one being the scaled target object. The achieved success rate is 57.89% for the push task, i.e., the percentage of executed instructions that correctly identified the target object and finished successfully. This result can be seen as a moderate generalization rate (substantially higher than random chance, i.e. 17%, but lower than the success rate in previous experiments). One possible explanation is the relatively large range of values for the scaling factor that we allowed for in this experiment, namely up to 3x times the original size.

1.4 Task Execution under Unseen Synonyms

In this experiment, we evaluate the ability of our model to identify the correct object even when a synonym is used. To this end, we replace object names by synonyms as shown in Tab. 9. As synonyms we utilise both single word and short phrase candidates. 10 synonyms are used for each object (we excluded the Pepsi can, since the same synonyms apply also to the Coke can) adding up to 50 synonyms in total. The achieved success rate for push task is 82.5%, which indicates a reasonably high degree of linguistic generalization.

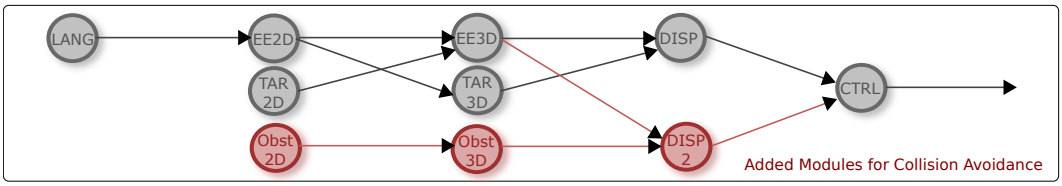

Figure 4: The new hierarchy of modules after adding the obstacle related modules. Obst2D locates the obstacle in the image. Obst3D predicts the obstacle spatial coordinates. DISP2 is the displacement between the end-effector and the obstacle.

1.5 Task Execution with Unseen Objects

In this experiment, we evaluate the ability of our model to recognize new, previously unseen objects. To this end, we create a new set test of 15 objects 6 of which are of similar object type as our training data (e.g. bottles, bread, etc.) and 9 are completely unseen object categories (e.g. helmet, horse, etc.). In the executed instructions we referred to the objects only by name in order to avoid identification through color. The resulting success rate of 34.6% for push task, does not indicate a high degree of generalization. However, this is to be expected since the model was trained on a small data set of geometries. Multiple possible remedies for this limitation exist, such as a.) training with a larger dataset, b.) using unsupervised pre-training, c.) leveraging existing vision backbones etc.

1.6 Obstacle Avoidance: Adding New Modules to the Hierarchy

In this experiment, we address the question "whether there is a clean way to add new elements to the hierarchy". More specifically, we investigate adding the capability to avoid obstacles (as suggested by the reviewer) on the way to performing a manipulation action. To this end, we first add new modules that detect the obstacle. This is done in the same vain as previously for the target or the end-effector. The resulting modules OBST2D and OBST3D are trained to generate the location of obstacle in image space and world space. More specifically, OBST2D identifies image patches that belong to the object. In turn, these patches are fed into OBST3D to generate a 3D world-coordinate.

Obstacle avoidance also involves the robot itself. Hence, we need to relate the detected obstacle position to the location of the robot. To this end, EE3D and OBST3D are used to calculate a second displacement DISP2 – this time between the obstacle and the end-effector. Fig. 4 shows the new hierarchy. The output of DISP2 feeds into the calculation of the control value where it is combined with the output of DISP (the displacement of the end-effector to the target object). All new modules are shown in red in Fig. 4.

For training the model, we introduced a basketball as an obstacle and placed it randomly within the workspace. Training trajectories that avoid the obstacle were generated by using a potential field approach [1]. More specifically, the basketball is a repulsor that pushes the end-effector away from it. Using this approach, we collected 200 training demonstrations. Note that in some demonstrations the basketball is visible but not in the way of the robot.

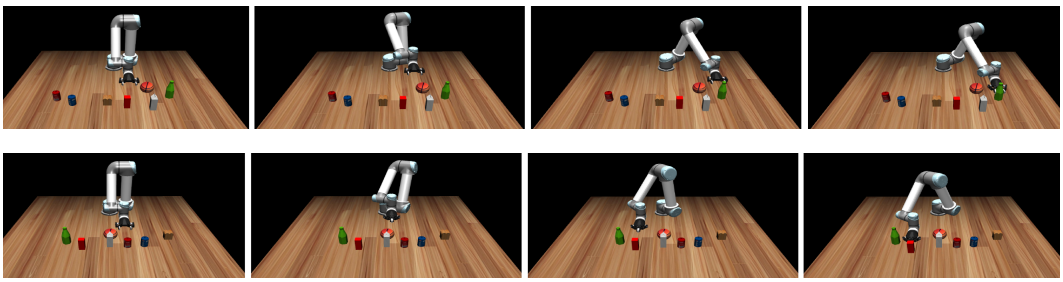

Figure 5: Robot performing a task while avoiding a basketball. The top row shows a pick action and the bottom row shows a push action. In both cases the robot changes its course to avoid collision with the obstacle.

To evaluate the new model, we performed a set of 100 test trials in which the obstacle was placed so as to block the robot's path. To this end, we compute the midpoint of the line connecting the robot to the target object and and apply a random perturbation. We define success as task completion without any collision between the ball and the robot. The achieved success rate for push task is 88%. In the cases where collisions occurred, we noticed that this was often the result of the robot fingertips touching the ball. A potential remedy would be to retrain EE3D module so as to focus on the fingertips of the gripper.

2 Training Details

2.1 Input Embeddings, Tokens and Register Slots

In our model, we use a multi-layer attention module for the interaction between different modalities. The input and output of an attention layer is a matrix respectively, which is composed of a sequence of embeddings $\boldsymbol{V} \in \mathbb{R}^{N \times E}$. There are N tokens and each token has the shape of $1 \times E$. Overall, the embedding $\boldsymbol{V}$ is the concatenation of the following tokens:

- **Vision tokens:** Vision tokens are the flattened output from a CNN. We use the CNN to downsample the raw input visual image from $224 \times 224 \times 3$ to $28 \times 28 \times 192$, and then flatten it to 784×192. Now, each token from this sequence has the shape of 1×192, and represents a patch of the original image.
- **Language tokens:** The language token is the output of CLIP model's language encoder. With the input being a natural language sentence, the language encoder generates a token of 1×500. We then downsize it to 1×192 as the language token.
- **Proprioception tokens:** Our model also takes joint angles as inputs. We simple use a multi-layer perceptron for the purpose of encoding joint angles. The raw input are 1×7 or 1×8, depending on the robot and gripper DoFs. We also transform that into a token of 1×192 as the input the attention layers.
- **Register slot tokens:** Register slots are used to store the output of a module so that it can be accessed in subsequent modules in the hierarchy. Accordingly, each module within our method has corresponding register slot tokens. The role of the register slots is to provide access to the output of previously executed modules within the hierarchy.

For processing attention, the register slot tokens are used as Queries, while all the tokens together are used as Keys and Values.

2.2 Supervision Labels

The supervision labels are used to focus attention on certain parts of the input embeddings. The attention focus depends on the underlying task. In the TAR2D and EE2D tasks, the supervision label is set to 1 for patches of the image that contain the target or end-effector respectively and to 0 otherwise. Fig. 6 depicts and explains the process of generating supervision labels for target prediction. The attention layer for module TAR2D should only focus on the image patch which includes the object. Accordingly, the label for this patch is set to one. Consequently, the TAR3D attention layer takes the output of TAR2D as input value and generates the world coordinate of the target as output. Therefore, the attention label for the register slot

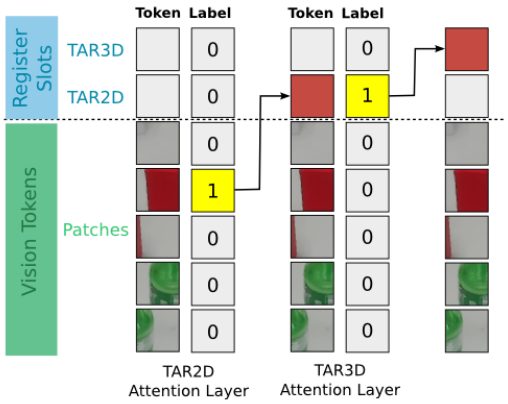

Figure 6: For TAR2D task, the supervision label is set to 1 for the image patch containing the target object. The output is then stored in the TAR2D register slot. In the next attention layer, for TAR3D task, the label is highlighted only for the TAR2D register slot, and the output is stored in TAR3D register slot.

of TAR2D is set to 1. The register slot for TAR3D will subsequently contain the 3D world coordinate of the object.

For the sake of completeness, below a list detailing the role and function of the supervision labels for each of the modules:

- **EE2D and TAR2D:** Given an input image embedding $\boldsymbol{I} \in \mathbb{R}^{M \times E}$ with M patches of dimension E, we require a binary matrix $\boldsymbol{B}$ of size $M \times 1$ as the attention supervision label. To this end, all patches in which the center of the end-effector or the object is located are set to 1. Each module is assigned a register slot shaped $1 \times E$ respectively, in which the output will be stored.
- **EE3D and TAR3D:** The two modules calculating 3D positions of the end-effector and target take as input only the output of the corresponding 2D modules, i.e., EE2D and TAR2D. To achieve this goal, we set the attention supervision label for the corresponding register slots to 1.
- **LANG and CTRL:** In the LANG task, the supervised attention label highlights the slot pertaining to the language embedding. In the CTRL task, the label highlights the slots for the language embedding, as well as the register slots for TAR3D and DISP (displacement between target and end-effector). This is due to the fact that the final control should take into account the target object, its relationship to the gripper and the linguistic instruction.

This cascaded processing of information is guided by the proposed supervised attention mechanism. Register slots play an integral part in the routing of information throughout the hierarchy. They store the outputs of individual modules for later access in subsequent modules. Supervised attention is used to force modules to access or neglect the information in register slots.

Table 1: Table of subtasks for modularity.

Abbreviation	Subtask Description
LANG	Get target object from language
TAR2D	Find object patch
TAR3D	Calculate object position
EE2D	Find robot end-effector patch
EE3D	Calculate end-effector position
DISP	Find distance object to end-effector
CTRL	Predict robot positions for control

2.3 Data Collection and Labelling

A simulated dataset was collected for all 3 robots involved in the experiments (Kinova, UR5 and Franka). We recorded 2000 demonstrations for training a policy from scratch, and 400 for transferring a policy to an unseen robot. For each of the demonstrations, we recorded object positions, robot proprioception (joint angle data), camera images, and end-effector positions throughout the whole trajectory at 125 Hz. As shown in Fig. 7, we perform a transformation of the positions into image patches, thereby locating the end-effector and objects in the image. Images are collected at a resolution of 224×224 resolution. Each demonstration typically comprises 100-400 timesteps.

For real-robot experiments, we recorded robot proprioception, camera images, end-effector position and the target object position. A higher cost comes from collecting 3D object positions. In order to calculate the 3D end-effector position, we use the forward kinematics of the robot and transform the result into camera coordinates. However, this process requires extrinsic and intrinsic camera information and therefore a manual calibration step. Similarly, in order to generate an estimate of the object's position, we use the robot end-effector's Tool Center Point (TCP) position after it touches the object. For higher accuracy, an object tracking algorithm [2] can also be used instead. The overall labeling process is **largely automated and only involves human intervention in two steps**, namely a.) extrinsic and intrinsic camera calibration and b.) which modules should attend to which register slots. The latter step (b) can also be automated if the structure of the hierarchy is given (as in Fig. 4) – attention is set to 1 for register slots of all modules that have inbound transitions

Figure 7: In order to collect labels for supervised attention in images, we firstly track the 3D position of the object, which is then projected onto the 2D image, and falls into one of the image patches. In the real world experiment, this patch along with the adjacent patches are selected as the labels.

Object	Adj	Noun
Coke	red coke cocacola	can bottle
Pepsi	blue pepsi pepsi coke	can bottle
Bottle	green glass ‘’ green glass	bottle
Carton	milk white	carton box
Cube	red maroon	object cube square
Bread	‘’	bread yellow object brown object

Table 2: The noun phrase template.

Verb Pick	Verb Push	Verb Put
pick pick up raise	push move	put down place down

Table 3: The verb phrase template.

Annotator Labeled Sentences	Success
Grab the loafs	F
put down the lime soda	T
lay down the red block	T
tip over the azure can	T
lift the white carton	F
knock over the pastry	T
lift the coke can	T
put down the sprite	T
grab the pepsi	T
elevate the red cube	T
Pick up the red cube	T
Lift up the blue cylinder	T
Move away the brown object	T
Push away the white object	T
Lift the blue object	T
Put down the green sprite	T
Push the green sprite	T
Push the reddish can	T
Pick up the milk container	F
Hold up the milk carton	F
Please pick up the green thing	F
Lift the red colored coke can	T
Push the yellow bread	T
Grab the blue colored can	T
Nudge that green bottle	F
Put down the red colored cuboid	T
Lift the white box	T
Take the pepsi off the table	F
Push the green object forward	F
Put down the zero coke on the desk	T

Table 4: Sentences collected from annotators for evaluation purposes. Our model achieves 73.3% success rate on variations of languages.

to the current layer/module. Also, the camera calibration step only has to be done once per robot setup.

For learning natural language instructions, we use a template to generate well-formed sentences during demonstrations. The template first randomly chooses a verb phrase according to Tab. 3, and then determines a noun phrase by randomly picking from Adj and Noun in Tab. 2. We leverage this procedure to generate sentences during training, validating and testing. In addition, we also,

for evaluation purposes, collected 30 natural language sentences from human annotators where our model achieves 73.3% success rate. The full list can be found in Tab. 4.

2.4 Network Architectures and Hyperparameters

We use a convolutional neural network for image encoding, as shown in Tab. 5. We use fully connected layers for joint encoders, target position decoders, displacement decoders and controllers, which are shown in Tab. 6, Tab. 7 and Tab. 8 respectively. We use 4 eight-head attention layers of 192 dimensions for modality fusing and interaction. The Adam optimizer with learning rate of 1e-4 is adopted for training.

Table 5: Image Encoder Architecture

Layer	Kernel	Channel	Stride	Padding
CNN	7	64	1	3
CNN	3	128	2	1
CNN	3	256	2	1
CNN	3	256	2	1
ResBlock	3	256	1	1
ResBlock	3	256	1	1
ResBlock	3	256	1	1

Table 6: Joint Encoder Architecture

Layer	Dimension
FC	256
FC	128
FC	192

Table 7: Position and Displacement Decoder Architecture

Layer	Dimension
FC	128
FC	9

Table 8: Controller Architecture

Layer	Dimension
FC	2048
FC	1024
FC	256
FC	120

Table 9: Synonyms Used in Test

Milk Carton	Bottle	Coke	Cube	Bread
skimmed milk package	soda	coke zero	brick	cinnamon roll
goat milk carton	Perrier	round container	block	sourdough
milk case	tonic	can	cuboid	brown bread
white packet	flask	coca cola	bar	loaf
milk parcel	pitcher	red soda	solid lump	naan
cream carton	container	cola	rectangular object	rye bread
cream package	decanter	metal container	solid piece	toast
heavy milk carton	vial	small soft drink	slab	gluten free food
almond milk box	vessel	fizzy drink	cuboidal slice	light bread
goat milk packs	cruet	diet coke	square object	food

References

[1] O. Khatib. The potential field approach and operational space formulation in robot control. In *Adaptive and learning systems*, pages 367–377. Springer, 1986.

[2] F. Chen, X. Wang, Y. Zhao, S. Lv, and X. Niu. Visual object tracking: A survey. *Computer Vision and Image Understanding*, page 103508, 2022.