NOIR: Neural Signal Operated Intelligent Robot for Everyday Activities: Appendix

Anonymous Author(s)
Affiliation
Address
email

1 Questions and Answers about NOIR

Q (Safety) : Is EEG safe to use? Are there any potential risks or side effects of using the EEG for extended periods of time?

A : EEG devices are generally safe with no known side effects and risks, especially when compared to invasive devices like implants. We use saline solution to lower electrical impedance and improve conductance. The solution could cause minor skin irritation when the net is used for extended periods of time, hence we mix the solution with baby shampoo to mitigate this.

Q (Safety) : How does the system ensure user safety, particularly in the context of real-world tasks with varying environments and unpredictable events?

A : We implement an EEG-controlled safety mechanism to confirm or interrupt robot actions with muscle tension, as decoded through clenching. Nevertheless, it is important to note that the current implementation entails a 500ms delay when interrupting robot actions which might lead to a potential risk in more dynamic tasks. With more training data using a shorter decoding window, the issue can be potentially mitigated.

Q (Universality) : Can EEG / NOIR be applied to different people? Given that the paper has only been tested on three human subjects, how can the authors justify the generalizability of the findings?

A : The EEG device employed in our research is versatile, catering to both adults and children as young as five years old. Accompanied by SensorNets of varying sizes, the device ensures compatibility with different head dimensions. Our decoding methods have been thoughtfully designed with diversity and inclusion in mind, drawing upon two prominent EEG signals: steady-state visually evoked potential and motor imagery. These signals have exhibited efficacy across a wide range of individuals. However, it is important to acknowledge that the interface of our system, NOIR, is exclusively visual in nature, rendering it unsuitable for individuals with severe visual impairments.

Q (Portability) : Can EEG be used outside the lab?

A : While mobile EEG devices offer portability, it is worth noting that they often exhibit a comparatively much lower signal-to-noise ratio. Various sources contribute to the noise present in EEG signals, including muscle movements, eye movements, power lines, and interference from other devices. These sources of noise exist in and outside of the lab; consequently, though we've chosen to implement robust decoding techniques based on classical statistics, more robust further filtering techniques to mitigate these unwanted artifacts and extract meaningful information accurately are needed for greater success in more chaotic environments.

Q (Privacy) : How does the system differentiate between intentional brain signals for task execution and other unrelated brain activity? How will you address potential issues of privacy and security?

A : The decoding algorithms employed in our study were purposefully engineered to exclusively capture task-relevant signals, ensuring the exclusion of any extraneous information. Adhering to the principles of data privacy and in compliance with the guidelines set by the Institutional Review Board (IRB) for human research, the data collected from participants during calibration and experimental sessions were promptly deleted following the conclusion of each experiment. Only the decoded signals, stripped of any identifying information, were retained for further analysis.

Submitted to the 7th Conference on Robot Learning (CoRL 2023). Do not distribute.

Property	gel-based EEG	dry EEG	MEG	fMRI	fNIRS	implant
Invasive?	No	No	No	No	No	Yes
Cost	similar	lower	higher	higher	varies	higher
Universality	similar	better	similar	similar	similar	worse
Setup time	longer	shorter	similar	similar	longer	longer
Signal-to-noise ratio	better	worse	-	-	-	better
Temporal resolution	similar	lower	similar	lower	lower	-
Spatial resolution	similar	lower	higher	higher	higher	-

Table 1: A comparison between brain recording devices, using our saline-based EEG device as the baseline. Note that the comparison is based on the average products that are available on the market for research, and does not account for specialized or customized devices. Universality considers whether the device can be used by the general population. For signal-to-noise ratio, MEG, fMRI, and fNIRS record different types of neural signals which are not directly comparable to EEG. For implants, the temporal and spatial resolution largely depends on the particular type of implant device used.

Q (Scalability) : How scalable is the robotics system? Can it be easily adapted to different robot platforms or expanded to accommodate a broader range of tasks beyond the 20 household activities tested?

A : Within the context of our study, two notable constraints are the speed of decoding and the availability of primitive skills. The former restricts the range of tasks to those that do not involve time-sensitive and dynamic interactions. However, the advancement in decoding accuracy and the reduction of the decoding window duration may eventually address this limitation. These improvements can potentially be achieved through the utilization of larger training datasets and the implementation of machine-learning-based decoding models, leveraging the high temporal resolution offered by EEG.

The development of a comprehensive library of primitive skills stands as a long-term objective in the field of robotics research. This entails creating a repertoire of fundamental abilities that can be adapted and combined to address new tasks. Additionally, our findings indicate that human users possess the ability to innovate and devise novel applications of existing skills to accomplish tasks, akin to the way humans employ tools.

Q (Potential impact) : How exactly do both individuals with and without disabilities benefit from this BRI system?

A : The potential applications of systems like NOIR in the future are vast and diverse. One significant area where these systems can have a profound impact is in assisting individuals with disabilities, particularly those with mobility-related impairments. By enabling these individuals to accomplish Activities of Daily Living and Instrumental Activities of Daily Living [1] tasks, such systems can greatly enhance their independence and overall quality of life. Currently, individuals without disabilities may initially find the BRI pipeline to have a learning curve, resulting in inefficiencies compared to their own performance in daily activities in their first few attempts. However, robot learning methods hold the promise of addressing these inefficiencies over time, and enable robots to help their users when needed.

2 Comparison between Different Brain Recording Devices

We use the EGI NetStation EEG system which uses rapid application 128-channel saline-based EGI SensorNets. Here we justify our choice of using non-invasive, saline-based EEG as the recording device for brain signals. A comparison of different brain reading devices (gel-based EEG, dry EEG, MEG, fMRI, fNIRS, implant) and their advantages and disadvantages are shown in Table 1, using our device as the baseline. Two noticeable alternatives are functional magnetic resonance imaging (fMRI) and invasive implants. fMRI measures the small changes in blood flow that occur with brain activity, which has a very high spatial resolution hence fine-grained information such as object categories and language [2] can be decoded from it. But fMRI suffers from low temporal resolution, and the recording device is extremely costly and cannot be used in daily scenarios. Brain implants have a very good signal-to-noise ratio and have great potential. However, the main concern is that it requires surgery to be applied, and health-related risks are not negligible.

Robot	Skill	Parameters
Franka	Reaching	6D goal pose in world
Franka	Picking	3D world pos to pick, gripper orientation (choose from 4)
Franka	Placing	3D world pos to place, gripper orientation (choose from 3)
Franka	Pushing	3D world pos to start pushing, axis of motion (choose from 3)
Franka	Wiping	3D world pos to start wiping
Franka	Drawing	3D world pos
Franka	Pouring	3D world pos, gripper orientation (choose from 3)
Franka	Pulling	3D world pos, gripper orientation (choose from 2), pull direction (choose from 2)
Franka	Grating	3D world pos
Tiago	Navigating	ID of pre-defined positions and poses
Tiago	Picking	ID of the object
Tiago	Placing	ID of the object
Tiago	Pouring	ID of the object
Tiago	Dropping	ID of object to drop the grasped object by

Table 2: Parameterized primitive skills for Franka and Tiago robots.

3 System Setup

Robot platform. The robot we use in our tabletop manipulation task is a standard Franka Emika robot arm with three RealSense cameras. For mobile manipulation, we use a Tiago++ model from PAL Robotics, with an omnidirectional base, two 7-degrees-of-freedom arms with parallel-yaw grippers, a 1-degree-of-freedom prismatic torso, two SICK LiDAR sensors (back and front of the base), and an ASUS Xtion RGB-D camera mounted on the robot's head, which can be controlled in yaw and pitch. All sensors and actuators are connected through the Robot Operating System, ROS [3]. The code runs on a laptop with an Nvidia GTX 1070 that sends the commands to the onboard robot computer to be executed.

Primitive skills list. A list of primitive skills along with their parameters can be found in Table 2, eight for Franka (16 tasks) and five for Tiago (four tasks). Human users can accomplish all 20 tasks, which are long-horizon and challenging, using these skills.

4 Task Definitions

For systematic evaluation of task success, we provide formal definitions of our tasks in the format of BEHAVIOR Domain Definition Language (BDDL) language [4, 5]. BDDL is a predicate logic-based language that establishes a symbolic state representation built on predefined, meaningful predicates grounded in physical states [5]. Each task is defined in BDDL as an initial and goal condition parametrizing sets of possible initial states and satisfactory goal states, as shown in Fig. 7.2, 7.2, and 7.2 at the end of the appendix. Compared to scene- or pose-specific definitions which are too restricted, BDDL is more intuitive to humans while providing concrete evaluation metrics for measuring task success.

5 Experimental Procedure

EEG device preparation. In our experiments, we use the 128-channel HydroCel Geodesic SensorNet from Magstim EGI, which has sponge tips in its electrode channels. Prior to experiments, the EEG net is soaked in a solution containing dissolved conductive salt (Potassium Chloride) and baby shampoo for 15 minutes. After the soaking, the net is worn by the experiment subject, and an impedance check is done. This impedance check entails ensuring that the impedance of each channel electrode is $\leq 50.0\,\mathrm{k}\Omega$, by using a syringe to add more conductive fluid between the electrodes and the scalp. We then carefully put on a shower cap to minimize the drying of conductive fluid over the course of the experiment.

Instructions to subjects. Before commencing the experiments, subjects are given instructions on how to execute the SSVEP, MI, and muscle tension (jaw-clenching) tasks. For SSVEP, they are instructed to simply focus on the flickering object of interest without getting distracted by the other objects on the screen. For MI, similar to datasets such as BCI Competition 2003 [6], and as per extensive literature review [7], we instruct subjects to either imagine continually bending their hands at the wrist (wrist dorsiflexion)

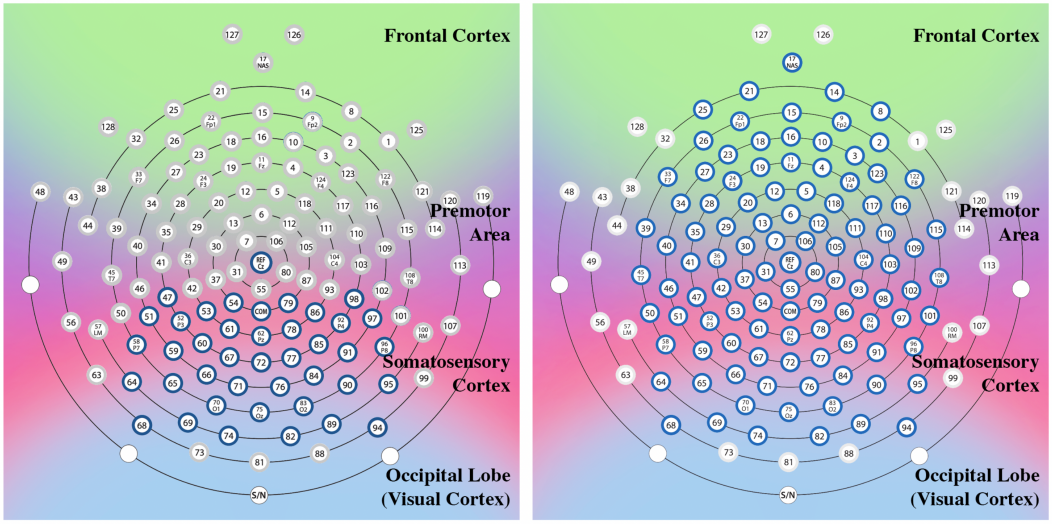


Figure 1: Map of relevant electrodes we use during SSVEP (Left) and Motor Imagery (Right).

or squeezing a ball for the hand actions ("Left", "Right"), and to imagine depressing a pedal with both feet (feet dorsiflexion) for the "Legs" action. For the "Rest" class, as is common practice in EEG experiments in general, we instruct users to focus on a fixation cross displayed on the screen. Subjects were told to stick with their actions of choice throughout the experiment, for consistency. For muscle tension, subjects were told to simply clench their jaw without too much or too little effort.

Interface. For SSVEP, subjects are told in writing on the screen to focus on the object of interest. Thereafter, a scene image of the objects with flickering masks overlaid on each object is presented, and we immediately begin recording the EEG Data over this period of time. For MI, the cues are different during calibration and task-time. During calibration, subjects are presented with a warning symbol (`.`) on screen for 1 second, before being presented with the symbol representing the action they are to imagine (`<-`: "Left", `->`: "Right", `v`: "Legs", `+`: "Rest"), which lasts on screen for 5500 ms. We record the latter 5000 ms of EEG data. After which, there is a randomized period of rest the lasts between 0.5 and 2 seconds, before the process repeats for another randomly chosen action class. This is done in 4 blocks of 5 trials per action, for a total of 20 trials per action. This procedure is again similar to datasets like BCI Competition 2003 [6], that use non-linguistic cues and randomization of rest / task. At task-time, similar to SSVEP, subjects are told in writing on the screen to perform MI to select a robot skill to execute. Thereafter, a written mapping of a class symbol ({`<-`, `->`, `v`, `+`}) to skill ({`pick_from_top`, `pick_from_side`, `...`}) is presented, and we begin recording EEG Data after a 2-second delay. For muscle tension, there is also a calibration phase, similar to MI, which entails collecting three 500ms-long trials for each class ("Rest", and "Clench") at the start of each experiment. The cues are written on the screen in words. At task time, when appropriate, written prompts are also presented on the screen (e.g. "clench if incorrect"), followed by a written countdown, after which the user has a 500ms window to clench (or not).

6 Decoding Algorithms Details

For both SSVEP and MI, we select a subset of channels and discard the signals from the rest, as shown in Figure 1. They correspond to the visual cortex for SSVEP, and the motor and visual areas for MI (with peripheral areas). For muscle tension (jaw clenching), we retain all channels.

SSVEP. To predict the object of interest, we apply Canonical Correlation Analysis (CCA) as shown in [8] to the collected SSVEP data. As each potential object of interest is flashing at a different frequency,

we are able to generate reference signals Y_{f_n} for each frequency f_n:

$$Y_{f_n} = \begin{bmatrix} \sin(2\pi f_n t) \\ \cos(2\pi f_n t) \\ \sin(4\pi f_n t) \\ \cos(4\pi f_n t) \end{bmatrix}, t = \begin{bmatrix} \frac{1}{f_s} & \frac{2}{f_s} & \dots & \frac{N_s}{f_s} \end{bmatrix} \tag{1}$$

where f_s is the sampling frequency and N_s is the number of samples.

Let X refer to the collected SSVEP data, and Y refer to a set of reference signals for a given frequency. The linear combinations of X and Y can be represented as $x = X^\top W_x$ and $y = Y^\top W_y$, and CCA finds the weights W_x and W_y that maximizes the correlation between x and y by solving the following equation:

$$\max_{W_x, W_y} \rho(x,y) = \frac{\mathbb{E}(W_x^\top X Y^\top W_y)}{\sqrt{\mathbb{E}(W_x^\top X X^\top W_x)\mathbb{E}(W_y^\top Y Y^\top W_y)}} \tag{2}$$

By calculating the maximum correlation ρ_{f_n} for each frequency f_n used for potential objects of interest, we are then able to predict the output class by finding $\text{argmax}_{f_n}(\rho_{f_n})$ and matching the result to the object of interest with that frequency.

Furthermore, we are able to return a list of predicted objects of interest in descending order of likelihood by matching each object to a list of descending maximum correlations ρ_{f_n}.

Motor imagery. To perform MI classification, we first band-pass filter the data between 8Hz - 30Hz, as that is the frequency range that includes the μ-band and β-band signals relevant to MI. The data is then transformed using the Common Spatial Pattern (CSP) algorithm. CSP is a linear transformation technique that applies a rotation to the data to orthogonalize the components where the over-timestep variance of the data differs the most across classes. We can then use the log-variance of each time series after rotation as features and perform QDA. Thereafter, we extract features by taking the normalized variance of this transformed data (called "CSP-space data"). We then perform Quadratic Discriminant Analysis (QDA) on this data. To calculate our calibration accuracy, we perform K-fold cross validation with $K_{\text{CV}} = 4$, but we use the entire calibrate dataset to fit the classifier for deployment at task-time.

CSP can be very briefly described as a process which orthogonalizes variance. To illustrate in the 2-class case, suppose the i-th calibration EEG time-series for class k can be written as $X_k^{(i)} \in \mathbb{R}^{C \times T}$, where $C =$ number of channels, T = number of time-steps, $i \in [1,20]$, and $k \in \{1,2\}$. Suppose further that the data is mean-normalized. Then:

$$\begin{aligned} \hat{\text{Cov}}(X_k) &= \frac{1}{20}\sum_{i=1}^{20} \text{Cov}\left(X_k^{(i)}\right) \\ &= \frac{1}{20}\sum_{i=1}^{20} \frac{1}{T} X_k^{(i)} X_k^{(i)^\top} \end{aligned} \tag{3}$$

And we perform a simultaneous diagonalization of $\{\hat{\text{Cov}}(X_k)\}$: $\text{Cov}(X_2)^{-1}\text{Cov}(X_1) = Q\Lambda Q^\top$. The transformation of any time-series X into the CSP-space is simply:

$$X_{\text{CSP}} = XQ \tag{4}$$

Note that we only keep the first $N_{\text{CSP}} = 4$ columns of XQ. The Python `mne` package [9] provides a multi-class generalization of this algorithm that we use. Feature extraction can be readily done by taking the component-wise variance of X_{CSP}, but we find that taking the normalized component-wise log-variance is better, as corroborated by previous studies [10]:

$$f_p(X) = \log\left(\frac{\text{Var}(X_{\text{CSP},p})}{\sum_{i=j}^{N_{\text{CSP}}} \text{Var}(X_{\text{CSP},j})}\right) \tag{5}$$

$$f(X) = (f_1(X), \dots, f_{N_{\text{CSP}}}(X)) \tag{6}$$

where $X_{\text{CSP},j}$ denotes the j-th column of X_{CSP}. The QDA step is straightforward: given our calibration dataset $\{f(X_k^{(i)})\}$, we simply fit a quadratic discriminant using the Python `sklearn` package, which allows us to recover a list of MI class predictions in decreasing order of likelihood.

Input dimension	2048
Number of hidden layers	5
Hidden layer dimension	1024
Output dimension	1024
Number of epochs	100
Batch size	40
Optimizer	Adam
Learning rate	0.001

Table 3: Feature embedding model and training hyperparameters for object and skill learning.

Muscle tension. To detect jaw clenches, electromyography (EMG) data is relevant. This is also picked up by our EEG net, and which, for succinctness, we will refer to as EEG data as well. Facial muscle tension results in a very significant high-variance signal across almost all channels that is very detectable using simple variance-based threshold filters without having to perform any frequency filters. Recall that we record three 500ms-long trials for each class ("Rest", "Clench"). In short, for each of the calibration time-series, we take the variance of the channel with the median variance; call this variance m. Then, we just take the mid-point between the maximum m between the rest samples, and the minimum m between the clench samples, and have this be our threshold variance level.

Let $X_k^{(i)} \in \mathbb{R}^{C\times T}$, where C = number of channels, T = number of time-steps, $i \in [1,3]$, and $k \in \{\text{Rest, Clench}\}$.

$$m_k^{(i)} = \underset{c}{\text{median}}\left\{\text{Var}\left(X_{k,c}^{(i)}\right)\right\},\ c \in [1,C] \tag{7}$$

where $X_{k,c}^{(i)}$ denotes the c-th row of $X_k^{(i)}$.

$$\text{Threshold} = \frac{1}{2}\left(\max_i\{m_{\text{Rest}}^{(i)}\} + \min_i\{m_{\text{Clench}}^{(i)}\}\right) \tag{8}$$

7 Robot Learning Algorithm Details

7.1 Object and skill learning details

We utilize pre-trained R3M as the feature extractor. Our training procedure aims to learn a latent representation of an input image for inferring the correct object-skill pair in the given scene. The feature embedding model is a fully-connected neural network that further encodes the outputs of the foundation model. Model parameters and training hyperparameters are summarized in Table 3. Collecting human data using a BRI system is expensive. To enable few-shot learning, the feature embedding model is trained using a triplet loss [11], which operates on three input vectors: anchor, positive (with the same label as the anchor), and negative (with a different label). Triplet loss pulls inputs with the same labels together by penalizing their distance in the latent space, as well as pushes inputs with different labels away. The loss function is defined as:

$$\mathcal{J}(a,p,n) = \max(\|f(a) - f(p)\|_2 - \|f(a) - f(n)\|_2 + \alpha,\, 0) \tag{9}$$

where a is the anchor vector, p the positive vector, n the negative vector, f the model, and $\alpha = 1$ is our separation margin.

Generalization test set. We test our algorithm in the following generalization settings:

Position and pose. For position generalization, we randomize the initial positions of all objects in the scene with fixed orientation and collect 20 different trajectories. For the pose generalization, we randomize both the initial positions and orientations of all objects.

Context. The context-level generalization refers to placing the target object in different environments, defined by different backgrounds, object orientations, and the inclusion of different objects in the scene. We collect 20 different trajectories with these variations.

Instance. The instance generalization aims to assess the model's capability to generalize across different types of objects present in the scene. For our target task (`MakePasta`), we collect 20 trajectories with 20 different kinds of pasta with different shapes, sizes, and colors.

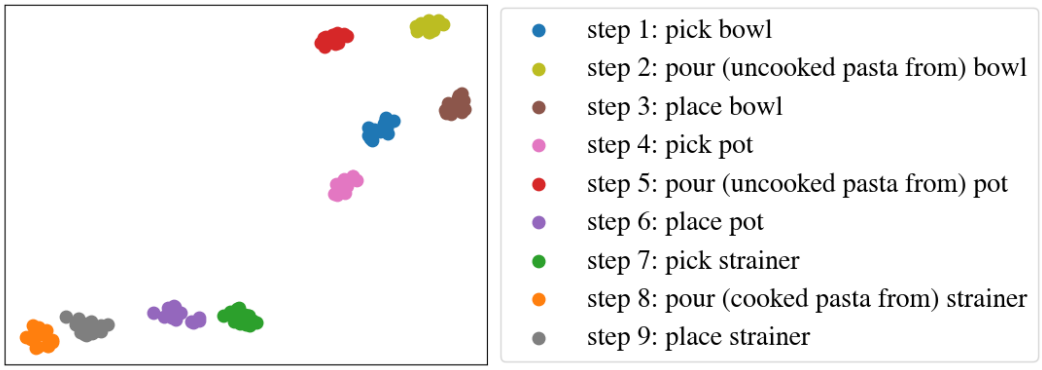


Figure 2: t-SNE visualization of latent representation generated by object and skill learning embedding model for `MakePasta` task pose generalization dataset.

Latent representation visualization. To understand the separability of latent representations generated by our object-skill learning model, we visualize the 1024-dimensional final image representations using the t-SNE data visualization technique [12]. Results for the `MakePasta` pose generalization test set are shown in Figure 2. The model can well separate each of the different stages of the task, allowing us to retrieve the correct object-skill pair for an unseen image.

7.2 One-shot parameter learning details

Design choices. We empirically found that using DINOv2's ViT-B model, alongside a 75x100 feature map and a 3x3 sliding window, with cosine similarity as the distance metric, resulted in the best performance for our image resolutions.

Generalization test set We test the generalization ability of our algorithm on 1008 unique training and test pairs, encompassing four types of generalizations including 8 position trials, 8 orientation trials, 32 context trials, and 960 instance trials.

Position and orientation The position and orientation generalizations, shown in Fig. 3 and Fig. 4 respectively, are tested in isolation, e.g. when the position is varied, the orientation is kept the same.

Context The context-level generalization, shown in Fig. 5, refers to placing the target object in different environments, e.g. the training image might show the target object in the kitchen while the test image shows the target object in a workspace. Here, we allow for position and orientation to vary as well.

Instance To test our algorithm's capability of instance-level generalization, shown in Fig. 6, we collected a set of four different object categories, each containing five unique object instances. Our object categories consist of mug, pen, bottle, and medicine bottle, whereas the bottle and medicine bottle categories consist of images from both the top-down and side views. We test all permutations within each object category including train and test pairs with different camera views. Here, we allow for position and orientation to vary as well.

Test set for comparing our method against baselines We test our method against baselines on 1080 unique training and test pairs, encompassing four types of generalizations including 8 position trials, 8 orientation trials, 32 context trials, 960 instance trials, 48 trials where we vary all four generalizations simultaneously, and 24 trials from the `SetTable` task.

Position, orientation, context, and instance simultaneously. Finally, we test our algorithm's ability to generalize when all four variables differ between the training and test image, shown in Fig. 7. Here, the only object category we use is a mug.

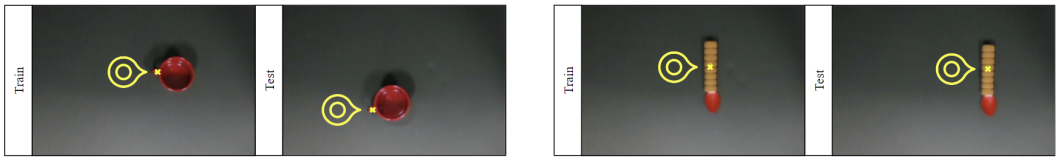


Figure 3: Position generalization. The first train parameter is set on the mug handle. The second train parameter is set on the spoon grip.

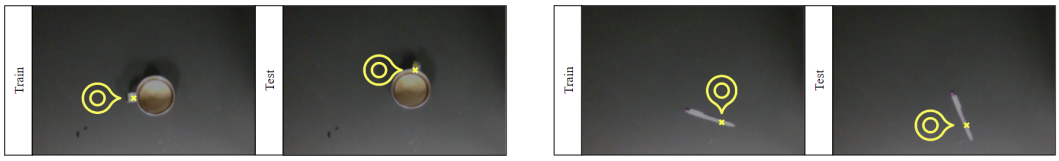


Figure 4: Orientation generalization. The first train parameter is set on the mug handle. The second train parameter is set on the pen grip.

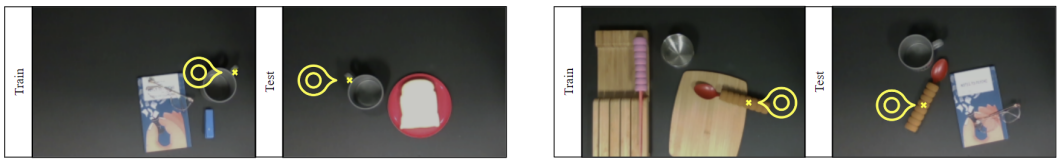


Figure 5: Context generalization. The first train parameter is set on the mug handle. The second train parameter is set on the spoon grip.

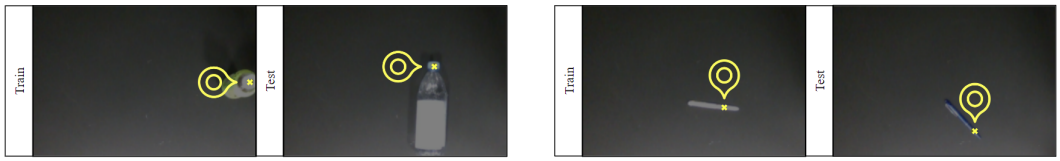


Figure 6: Instance generalization. First pair shows instance generalization with different camera views: from the top and from the side. The first train parameter is set on the bottle cap. The second train parameter is set on the pen grip.

References

[1] S. Katz. Assessing self-maintenance: activities of daily living, mobility, and instrumental activities of daily living. *Journal of the American Geriatrics Society*, 1983.

[2] J. Tang, A. LeBel, S. Jain, and A. G. Huth. Semantic reconstruction of continuous language from non-invasive brain recordings. *Nature Neuroscience*, pages 1–9, 2023.

[3] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, A. Y. Ng, et al. Ros: an open-source robot operating system. In *ICRA workshop on open source software*, volume 3, page 5. Kobe, Japan, 2009.

[4] C. Li, R. Zhang, J. Wong, C. Gokmen, S. Srivastava, R. Martín-Martín, C. Wang, G. Levine, M. Lingelbach, J. Sun, et al. Behavior-1k: A benchmark for embodied ai with 1,000 everyday activities and realistic simulation. In *Conference on Robot Learning*, pages 80–93. PMLR, 2023.

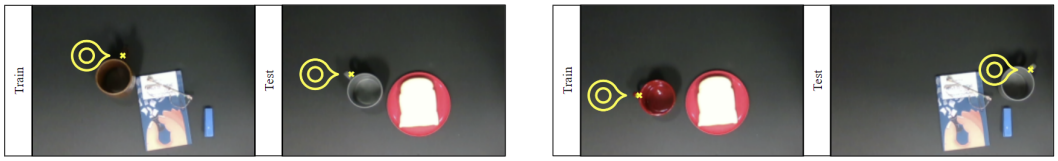


Figure 7: Position, orientation, instance, and context generalization. Both train parameters are set on the mug handle.

[5] S. Srivastava, C. Li, M. Lingelbach, R. Martín-Martín, F. Xia, K. E. Vainio, Z. Lian, C. Gokmen, S. Buch, K. Liu, et al. Behavior: Benchmark for everyday household activities in virtual, interactive, and ecological environments. In *Conference on Robot Learning*, pages 477–490. PMLR, 2022.

[6] B. Blankertz, K.-R. Muller, G. Curio, T. M. Vaughan, G. Schalk, J. R. Wolpaw, A. Schlogl, C. Neuper, G. Pfurtscheller, T. Hinterberger, et al. The bci competition 2003: progress and perspectives in detection and discrimination of eeg single trials. *IEEE transactions on biomedical engineering*, 51 (6):1044–1051, 2004.

[7] R. Scherer and C. Vidaurre. Motor imagery based brain–computer interfaces. In *Smart Wheelchairs and Brain-Computer Interfaces*, pages 171–195. Elsevier, 2018.

[8] A. Ravi, N. H. Beni, J. Manuel, and N. Jiang. Comparing user-dependent and user-independent training of cnn for ssvep bci. In *Journal of Neural Engineering*, 2020.

[9] A. Gramfort, M. Luessi, E. Larson, D. A. Engemann, D. Strohmeier, C. Brodbeck, R. Goj, M. Jas, T. Brooks, L. Parkkonen, et al. Meg and eeg data analysis with mne-python. *Frontiers in neuroscience*, page 267, 2013.

[10] H. Ramoser, J. Muller-Gerking, and G. Pfurtscheller. Optimal spatial filtering of single trial eeg during imagined hand movement. *IEEE transactions on rehabilitation engineering*, 8(4):441–446, 2000.

[11] K. Q. Weinberger and L. K. Saul. Distance metric learning for large margin nearest neighbor classification. *Journal of Machine Learning Research*, 10(9):207–244, 2009. URL http://jmlr.org/papers/v10/weinberger09a.html.

[12] L. van der Maaten and G. Hinton. Viualizing data using t-sne. *Journal of Machine Learning Research*, 9:2579–2605, 11 2008.

Sukiyaki

Initial	Goal
	

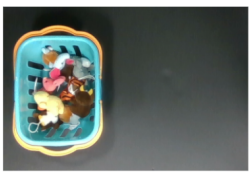
Objects	Initial Condition	Final Condition
Table Hotpot PlateA PlateB Saucepan Box Bowl Power switch Meat Tofu Shirataki Shiitake Shungiku Negi Broth Spatula	ontop(hotpot, table) ontop(bowl, table) ontop(plateA, table) ontop(plateB, table) ontop(sauce pan, table) ontop(power switch, table) ontop(box, table) ontop(meat, plateA) ontop(negi, plateA) ontop(shungiku, plateB) ontop(shiitake, plateB) inside(tofu, bowl) inside(shirataki, bowl) inside(broth, sauce pan) turned_off(power switch) in_hand(spatula)	ontop(hotpot, table) ontop(bowl, table) ontop(plateA, table) ontop(plateB, table) ontop(sauce pan, table) ontop(power switch, table) ontop(box, table) inside(meat, hotpot) inside(shungiku, hotpot) inside(shiitake, hotpot) inside(tofu, hotpot) inside(shirataki, hotpot) inside(broth, hotpot) turned_on(power switch) inside(spatula, box)

WipeSpill

Initial	Goal
	

Objects	Initial Condition	Final Condition
Table Cup Napkins Liquid	ontop(cup, table) inside(napkins, cup) covered(tabletop, liquid)	ontop(cup, table) not_covered(tabletop, liquid)

CollectToy

Initial	Goal
	

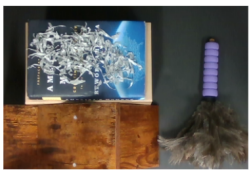
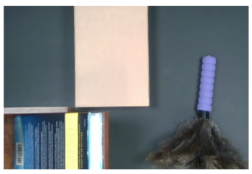
Objects	Initial Condition	Final Condition
Table Basket Toys ToyA~C	ontop(basket, table) ontop(ToyA~C, table) inside(Toys, basket)	ontop(basket, table) inside(ToyA~C, basket) inside(Toys, basket)

SweepTrash

Initial	Goal
	

Objects	Initial Condition	Final Condition
Table Dustpan Broom Trash	ontop(broom, table) ontop(trash, table) ontop(dustpan, table)	ontop(dustpan, table) in(trash, dustpan)

CleanBook

Initial	Goal
	

Objects	Initial Condition	Final Condition
Table Book shelf Book Dust Featherduster	ontop(book shelf, table) ontop(featherduster, table) ontop(book, table) covered(book, dust)	ontop(book shelf, table) ontop(book, book shelf) not_covered(book, dust)

IronCloth

Initial	Goal
	

Objects	Initial Condition	Final Condition
Table Iron Cloth Ironing board	ontop(ironing board, table) ontop(iron, table) ontop(cloth, ironing board) covered(cloth, wrinkles)	ontop(ironing board, table) ontop(cloth, ironing board) not_covered(cloth, wrinkles)

OpenBasket

Initial	Goal
	

Objects	Initial Condition	Final Condition
Table Basket Cloth Banana	ontop(basket, table) inside(banana, basket) ontop(cloth, basket)	ontop(cloth, table) ontop(basket, table) ontop(banana, table)

PourTea

Initial	Goal
	

Objects	Initial Condition	Final Condition
Table Cup Teabag Teapot Water Tea Plate	ontop(cup, table) ontop(plate, table) ontop(teabag, plate) ontop(teapot, table) inside(water, teapot)	ontop(cup, table) ontop(plate, table) inside(teabag, teapot) inside(tea, cup)

SetTable

Initial	Goal
	

Objects	Initial Condition	Final Condition
Table Lunch mat Container Bread Teapot Bowl Cup Spoon Water	ontop(lunch mat, table) ontop(container, table) ontop(teapot, table) ontop(plate, lunch mat) ontop(cup, table) ontop(spoon, plate) inside(bread, container) inside(water, teapot)	ontop(lunch mat, table) ontop(container, table) ontop(plate, lunch mat) ontop(cup, lunch mat) ontop(spoon, lunch mat) ontop(bread, plate) ontop(cup, lunch mat) inside(water, cup)

GrateCheese

Initial	Goal
	

Objects	Initial Condition	Final Condition
Table Container Grater Bowl Cheese block Grated cheese Plate Pasta	ontop(plate, table) ontop(cheese block, plate) ontop(container, table) inside(pasta, container) ontop(bowl, table) ontop(grater, bowl) absent(grated cheese)	ontop(container, table) inside(pasta, container) ontop(grated cheese, pasta)

CutBanana

Initial	Goal
	

Objects	Initial Condition	Final Condition
Table Cutting board Banana Knife Knife holder	ontop(cutting board, table) ontop(knife holder, table) ontop(banana, cutting board) inserted(knife, knife holder) not_sliced*(banana)	ontop(cutting board, table) ontop(knife holder, table) ontop(banana, cutting board) sliced**(banana)

*not_sliced = has no cuts, **sliced = has 3 parallel cuts

CookPasta

Initial	Goal
	

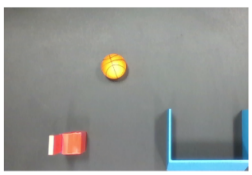
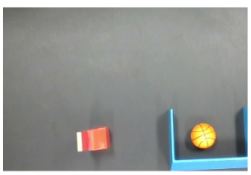
Objects	Initial Condition	Final Condition
Table Pasta BowlS BowlL Stove Pot Strainer Water Pitcher	ontop(stove, table) ontop(pot, stove) ontop(bowlS, table) ontop(bowlL, table) ontop(strainer, bowlL) inside(pasta, bowlS) inside(water, pitcher) not_cooked(pasta)	ontop(stove, table) ontop(bowlS, table) ontop(bowlL, table) ontop(strainer, bowlL) inside(pasta, strainer) cooked(pasta)

Sandwich

Initial	Goal
	

Objects	Initial Condition	Final Condition
Table ContainerA ContainerB Lid Bread1 Bread2 Tomato Lettuce Plate	ontop(containerA, table) ontop(containerB, table) ontop(plate, table) ontop(tomato, plate) ontop(lettuce, plate) inside(bread1, containerA) inside(bread2, containerA)	ontop(containerB, table) ontop(plate, table) inside(bread1, containerB) inside(tomato, containerB) inside(lettuce, containerB) inside(bread2, containerB) ontop(tomato, bread1) ontop(lettuce, tomato) ontop(bread2, lettuce)

Hockey

Initial	Goal
	

Objects	Initial Condition	Final Condition
Table Ball Goal Tool	ontop(ball, table) ontop(tool, table) ontop(goal, table) not_in(ball, goal)	ontop(goal, table) in(ball, goal)

OpenGift

Initial	Goal
	

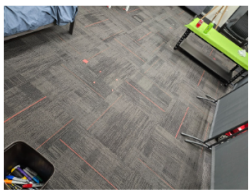
Objects	Initial Condition	Final Condition
Table Gift box Lid Card Ribbon Gift	ontop(gift box, table) ontop(lid, gift box) attached(card, lid) tied(ribbon, gift box) tied(ribbon, lid) tied(ribbon, card) inside(gift, gift box)	ontop(gift, table)

TicTacToe

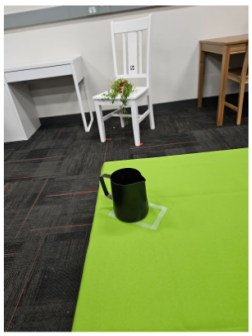
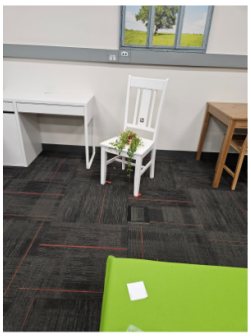
Initial	Goal
	

Objects	Initial Condition	Final Condition
Table Eraser Squares1~9 Marker "O" "X"	ontop(marker, table) empty*(squares2~9)	filled**(squares1~9) OR gameset***() **Followed by:** empty*(squares1~9)

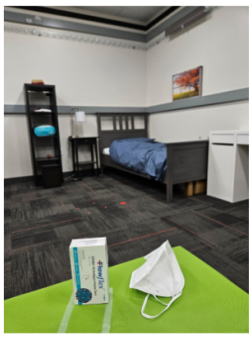
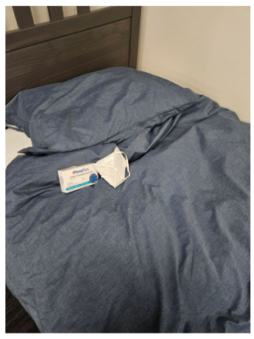
*empty = no "O" or "X" drawn, **filled = exactly 1 "O" or "X" drawn, ***there is a winner

TrashDisposal	
Initial	Goal
	

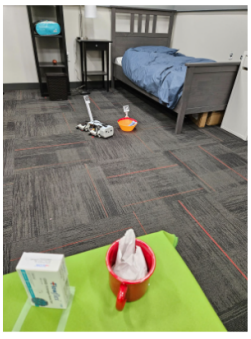
Objects	Initial Condition	Final Condition
Table Trashcan Can Bottle	ontop(can, table) ontop(bottle, table) ontop(trashcan, floor) inroom(table) inroom(trashcan)	inside(can, trashcan) inside(bottle, trashcan)

WaterPlant	
Initial	Goal
	

Objects	Initial Condition	Final Condition
Table Pitcher Bottle Plant Chair	ontop(bottle, table) ontop(pitcher, table) ontop(plant, chair) inroom(chair) inroom(table) dry(plant) dry(chair) dry(floor)	not_dry(plant) dry(chair) dry(floor)

CovidCare	
Initial	Goal
	

Objects	Initial Condition	Final Condition
Table Mask Testkit Bed	ontop(mask, table) ontop(testkit, table) inroom(table) inroom(bed)	ontop(mask, bed) ontop(testkit, bed)

PetDog	
Initial	Goal
	

Objects	Initial Condition	Final Condition
Table Bowl Dog Dogbone Testkit	inroom(table) inroom(bowl) inroom(dog) ontop(dogbone, table) ontop(testkit, table) not_petted(dog)	inside(bone, bowl) petted(dog)