SYNAPSE: Learning Preferential Concepts from Visual Demonstrations

Sadanand Modak[1], Noah Patton[1], Isil Dillig[1], Joydeep Biswas[1]

***Abstract*—We address the problem of *preference learning*, which aims to learn user-specific preferences (e.g., *"good parking spot"*, *"convenient drop-off location"*) from visual input. Despite its similarity to learning *factual* concepts (*e.g.*, "red cube"), preference learning is a fundamentally harder problem due to its subjective nature and the paucity of person-specific training data. We address this problem using a new framework called SYNAPSE, which is a neuro-symbolic approach designed to efficiently learn preferential concepts from limited demonstrations. SYNAPSE represents preferences as neuro-symbolic programs in a domain-specific language (DSL) that operates over images, and leverages a novel combination of visual parsing, large language models, and program synthesis to learn programs representing individual preferences. We evaluate SYNAPSE focusing on mobility-related concepts in mobile robotics and autonomous driving. Our evaluation demonstrates that SYNAPSE significantly outperforms existing baselines. The code and other details can be found on the project website https://amrl.cs.utexas.edu/synapse.**

I. INTRODUCTION

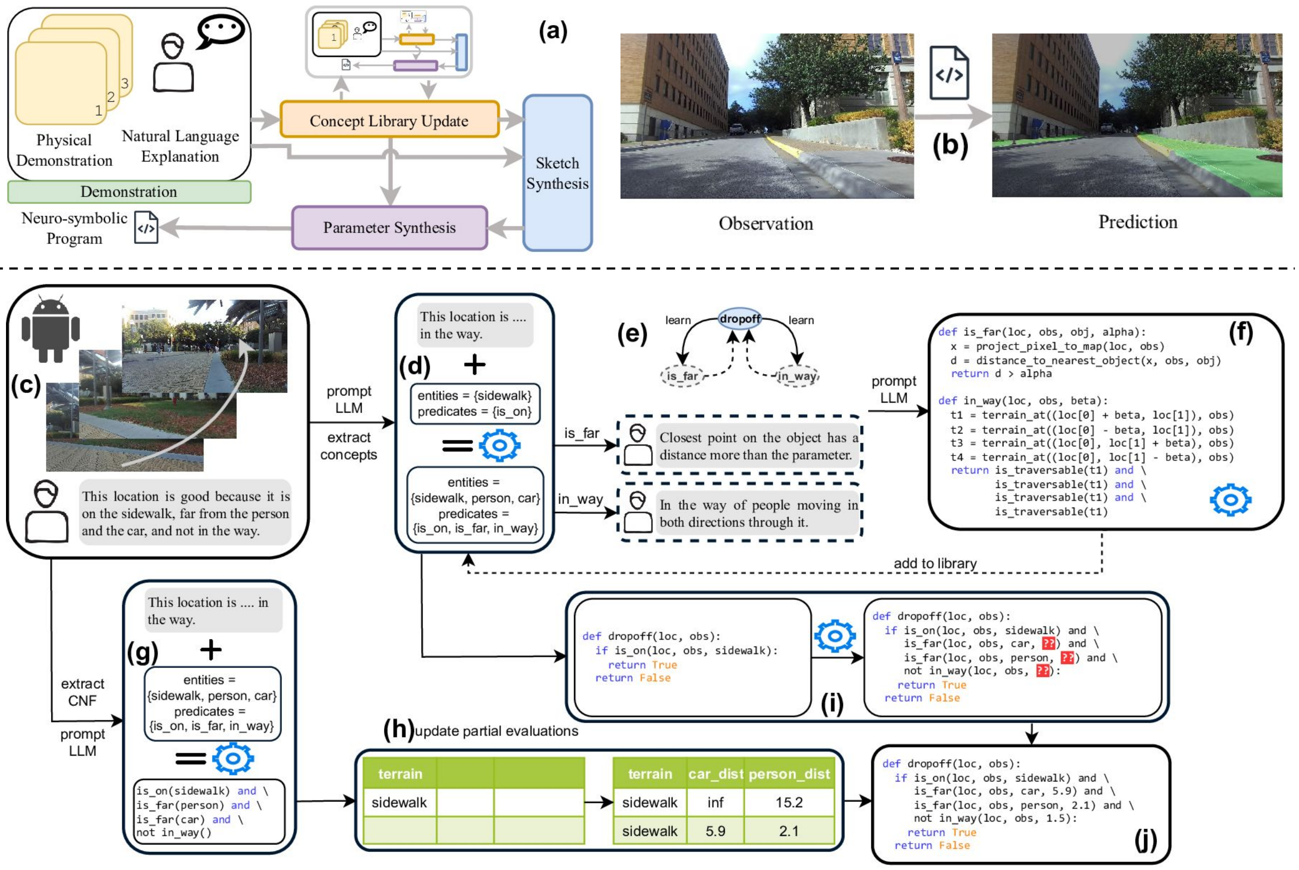
Imagine trying to come up with a definition of *"a good taxi drop-off location"*. One person may consider a spot to be a good drop-off location depending on whether it is close to the door of a building, while someone else might want it in the shade. Such concepts vary from person to person and inherently depend on their preferences. We call them *preferential concepts*, and we are interested in the problem of *preference learning* from visual input. Learning preferences is important because we want systems that are customizable and that can adapt to end-users. This problem is quite related to the task of visual concept learning, wherein much of the work focuses on learning concepts such as *having the color red* or *being to the left of another object* [1]–[16]. All such prior work assumes there is a *ground-truth* for the concept, i.e., the definition of the concept does not differ among people, and as a consequence, sufficiently many examples are available, and can be objectively evaluated. We refer to such concepts as *factual concepts*. While most prior work that learns visual concepts exploits the availability of large datasets such as CLEVR [17], those methods cannot be applied to preference learning because it is a data-impoverished setting by its very nature: a single individual can put up with providing only so much data. This limitation is also present in most of the preference learning work in the reinforcement learning literature as well, where human preferences are represented as neural networks [18], [19] or latent reward models [20]–[25]. Furthermore, because preferences are inherently individual, they can depend on entirely different concepts, such as in the drop-off location example above (i.e., based on *proximity to door* as opposed to *being in the shade*). This requires learning novel visual concepts in a *hierarchical* manner. Lastly, coming up with a complete definition of a preferential concept at once is itself a hard problem: it is much easier for someone to *show* examples that satisfy their intuition as humans tend to *build* their notion of a preferential concept over time. Thus, preference learning calls for an approach that can handle *incremental learning* from visual demonstrations.

To address these challenges, we present SYNAPSE, a novel framework that learns human preferences in a data-efficient manner. Figure 1 shows a schematic of our proposed SYNAPSE framework. In contrast to prior preference learning approaches [18]–[27] which take in weak reward signals to learn preferences, we use a more direct form of a preference signal, which consists of a physical demonstration including visual data, and a natural language (NL) explanation for the preference. Building up on recent works using LLMs to generate Language Model Programs (LMPs) [14], [16], [28]–[35], we use NL input from the human to identify new concepts to be learned as well as how to compose them in the form of an LMP. However, in addition to learning new concepts or composing existing ones, preferences also have a quantitative aspect. For example, to be a good drop-off spot, you should be close to a door, but exactly how close is a personal preference. This is where the demonstrations come into play and allow us to infer quantitative aspects of the preference that are hard to capture via natural language alone. Finally, to allow *incremental*, *data-efficient* learning, SYNAPSE expresses preferential concepts as programs in *neuro-symbolic* domain specific language (DSL) operating over images, and learns these programs based on demonstrations using constrained optimization techniques (based on *maximum satisfiability* [36]). Such a programmatic representation also facilitates *life-long learning*, allowing incremental changes to the learnt program as new demonstrations arrive.

To demonstrate the effectiveness of our framework, we evaluate it on three mobility-related visual preferential concepts which find their use in mobile robotics and autonomous driving. Empirical results show that SYNAPSE outperforms the baselines by a significant margin, especially when evaluated on out-of-distribution data — even when SYNAPSE is trained on an order of magnitude fewer examples than baseline Neural-Network (NN) approaches.

*This work is supported in part by NSF awards OIA-2219236, CCF-1762299, CCF-1918889, CNS-1908304, CCF-1901376, CNS-2120696, CCF-2210831, and CCF-2319471, as well as Amazon and JP Morgan

[1]Department of Computer Science, University of Texas at Austin, {smodak11, npatt, isil, joydeepb}@cs.utexas.edu



(a)
Physical Demonstration
Natural Language Explanation
Demonstration
Concept Library Update
Sketch Synthesis
Parameter Synthesis
Neuro-symbolic Program
(b)
Observation
Prediction
(c)
This location is good because it is on the sidewalk, far from the person and the car, and not in the way.
prompt LLM
extract concepts
(d)
This location is .... in the way.
entities = {sidewalk}
predicates = {is_on}
entities = {sidewalk, person, car}
predicates = {is_on, is_far, in_way}
is_far
in_way
(e)
learn
dropoff
learn
is_far
in_way
Closest point on the object has a distance more than the parameter.
In the way of people moving in both directions through it.
prompt LLM
(f)
add to library
extract CNF
prompt LLM
(g)
This location is .... in the way.
entities = {sidewalk, person, car}
predicates = {is_on, is_far, in_way}
(h) update partial evaluations
(i)
(j)

```
def is_far(loc, obs, obj, alpha):
  x = project_pixel_to_map(loc, obs)
  d = distance_to_nearest_object(x, obs, obj)
  return d > alpha

def in_way(loc, obs, beta):
  t1 = terrain_at((loc[0] + beta, loc[1]), obs)
  t2 = terrain_at((loc[0] - beta, loc[1]), obs)
  t3 = terrain_at((loc[0], loc[1] + beta), obs)
  t4 = terrain_at((loc[0], loc[1] - beta), obs)
  return is_traversable(t1) and \
         is_traversable(t1) and \
         is_traversable(t1) and \
         is_traversable(t1)
```

```
is_on(sidewalk) and \
is_far(person) and \
is_far(car) and \
not in_way()
```

terrain		
sidewalk		

terrain	car_dist	person_dist
sidewalk	inf	15.2
sidewalk	5.9	2.1

```
def dropoff(loc, obs):
  if is_on(loc, obs, sidewalk):
    return True
  return False
```

```
def dropoff(loc, obs):
  if is_on(loc, obs, sidewalk) and \
     is_far(loc, obs, car, ??) and \
     is_far(loc, obs, person, ??) and \
     not in_way(loc, obs, ??):
    return True
  return False
```

```
def dropoff(loc, obs):
  if is_on(loc, obs, sidewalk) and \
     is_far(loc, obs, car, 5.9) and \
     is_far(loc, obs, person, 2.1) and \
     not in_way(loc, obs, 1.5):
    return True
  return False
```

Fig. 1: Illustration of SYNAPSE for identifying good taxi drop-off locations. (a) SYNAPSE learns a neuro-symbolic program that represents the preferential concept based on user demonstrations, which include both a NL explanation and a physical demonstration. The learning algorithm consists of three steps, namely updating the concept library, synthesizing a program sketch, and performing parameter synthesis; (b) SYNAPSE evaluates the program on a new query image to return a preference (in this case, boolean) mask over the input image; (c) given a user demonstration, (d) we extract entities of relevance from the NL input; (e) for the new observed predicates, SYNAPSE recursively calls itself to first learn these auxiliary predicates from new user demonstrations (here, in the example, it takes the form of NL-only demonstrations) followed by (f) code generation; (g) once all the newly seen predicates are learned, a compact form representation (*i.e.*, CNF) is extracted from the NL input, and (h) partial evaluations are carried out from the physical demonstrations; (i) using the previous program sketch and the current CNF, a new sketch is synthesized; (j) finally, parameter synthesis fills the holes in the sketch using the partial evaluations.

II. METHOD

We define a *preference task* $\mathscr{T} := \langle O, Q, P \rangle$ as a tuple consisting of an observation space O, a query space Q, and a preference space P. A preference evaluator π accepts an observation and a query, and returns a preference value: $\pi : O \times Q \rightarrow P$. The goal of *preference learning* is to synthesize a suitable evaluator π that accurately predicts a person's visual preferences given an image. As stated earlier, a distinguishing feature of preference learning is that it must be performed using *small* amounts of training data due to the subjective nature of preferences. To enable data-efficient learning, our proposed SYNAPSE approach represents the preference evaluator π as a neuro-symbolic program in a DSL and synthesizes π from a small number of user demonstrations, where each demonstration includes a sequence of images along with a natural language explanation for the user's preference. In the following discussion, we first present our DSL for representing preferences and then describe our learning algorithm.

Algorithm 1: SYNAPSE-Learn

```
Input: a set of previously seen demonstrations $\mathscr{D}$, the new demonstration $d_n$,
  a potentially empty previous sketch $\hat{\pi}_o$, a previous concept library $\mathscr{C}$
Output: The new demonstrations set $\mathscr{D}'$, a neuro-symbolic preference
  evaluator $\pi$ parameterized by the new concept library $\mathscr{C}'$, the sketch $\hat{\pi}$ used
  to generate $\pi$
1: Learn($\mathscr{D}, d_n, \hat{\pi}_o, \mathscr{C}$)
2:     # Update concept library with new natural language utterance
3:     $\mathscr{C}' \leftarrow$ UpdateConceptLibrary($d_n.e, \mathscr{C}$)
4:     # Get the updated sketch from the new demonstration and previous sketch
5:     $\hat{\pi} \leftarrow$ SketchSynth($d_n, \hat{\pi}_o, \mathscr{C}'$)
6:     # Fill the holes in $\hat{\pi}$ based off of the demonstrations $\mathscr{D}'$
7:     $\pi \leftarrow$ ParamSynth($\hat{\pi}, \mathscr{D} \cup \{d_n\}$)
8:     return $\mathscr{D} \cup \{d_n\}, \pi, \mathscr{C}', \hat{\pi}$
```

A. Representing Preferences

We represent preference evaluation functions π (or preferences for short) in the neuro-symbolic DSL shown in Figure 2. This DSL is parameterized over a so-called *concept library*

Inputs

$$\begin{array}{lcl} q & \in & Q \\ o & \in & O \end{array}$$

Constants

$$\begin{array}{lcl} v & \in & \{\text{Int}, \text{Real}, \ldots\} \\ p & \in & P \end{array}$$

Terms

$$\begin{array}{l} t := q \mid o \mid v \\ \quad \mid f(t_1, \ldots, t_n) \ \text{ where } f \in \mathscr{C}_f \end{array}$$

Conditions

$$\begin{array}{lcl} \phi & := & p_c(t_1, \ldots, t_n) \text{ where } p_c \in \mathscr{C}_b \\ & & \mid \neg\phi \mid \phi \wedge \phi \mid \phi \vee \phi \end{array}$$

Programs

$$\pi \ := \ p \mid \texttt{if } (\phi) \texttt{ then } \pi \texttt{ else } \pi'$$

Fig. 2: The SYNAPSE neuro-symbolic DSL parametrized over concept library $\mathscr{C}$, which consists of a set of predicates $\mathscr{C}_b$ and functions $\mathscr{C}_f$.

$\mathscr{C}$, which includes both predicates $\mathscr{C}_b$ as well as non-boolean functions $\mathscr{C}_f$. The concept library includes built-in operators and predicates (e.g., $+, \times, \leq, \ldots$), pre-trained neural networks (e.g., for object classification and terrain detection), zero-shot visual language models (VLMs), as well as previously learned concepts and functions (expressed in the same DSL). We sometimes use the notation $\pi_{\mathscr{C}}$ to denote programs using concept library $\mathscr{C}$ and omit the subscript $\mathscr{C}$ where it is clear from context.

At a high level, a program π consists of (nested) if-then-else statements and is therefore conceptually similar to a decision tree. Each leaf of this decision tree is a preference (e.g., *good, bad, neutral*) drawn from the preference space P, which is assumed to be a finite set. Internal nodes of the decision tree are neuro-symbolic conditions ϕ, which include boolean combinations of predicates of the form $p(t_1, \ldots, t_n)$ where each t_i is a neuro-symbolic term and p is a predicate drawn from $\mathscr{C}_b$, which could be a built-in relation (e.g., $\leq$), result of a neural classifier, or a previously-learned concept (e.g., *close-to*).

B. Learning Preferences

We now discuss our learning algorithm, `SYNAPSE-Learn`, for synthesizing preference evaluation functions from a set of demonstrations $\mathscr{D}$. As `SYNAPSE-Learn` is meant to be used in a life-long-learning setting, we present it as an incremental algorithm that takes one additional demonstration in each invocation and returns an updated preference evaluation function. As mentioned earlier, we represent each demonstration d as a pair (t, e) where t is a physical demonstration consisting of a sequence of images and LiDAR point clouds and e is a natural language explanation for the preference. Given a demonstration d, we write $d.t$ and $d.e$ to denote its physical demonstration and explanation component respectively.

In addition to the new demonstration d_n, `SYNAPSE-Learn` takes three additional arguments, namely the previous set of demonstrations $\mathscr{D}$, the previously learnt preference evaluation function π_o (`None` for the first invocation), and the current concept library $\mathscr{C}$, which is initialized to contain only a set of built-in concepts. `SYNAPSE-Learn` uses the old program π_o to bootstrap the learning process, and the previous demonstrations are required to ensure that the updated program is consistent with *all* demonstrations provided thus far.

At a high level, the learning procedure consists of three steps, which are explained in more detail in the remainder of this section:

1) **Updating the concept library:** SYNAPSE first checks whether the existing concept library $\mathscr{C}$ is sufficient for successfully learning the desired preference evaluation function. For example, if the natural language explanation uses the term "far away" but the concept library does not contain a suitable definition, `SYNAPSE-Learn` interactively queries the user for clarification and updates its concept library as needed.
2) **Synthesizing a program sketch:** If the concept library is sufficient for representing the preference, `SYNAPSE-Learn` proceeds to synthesize a so-called *program sketch*, which is a program with missing constants to be synthesized. We differentiate between program sketches and complete programs because the user's natural language explanation is often sufficient to understand the general structure of the preference evaluation function but not its numeric parameters, which can only be accurately learned from the physical demonstrations. Thus, `SYNAPSE-Learn` generates the program sketch based only on the natural language explanation.
3) **Parameter synthesis:** The final phase of the learning algorithm utilizes all physical demonstrations provided thus far to synthesize the unknown numeric parameters of the sketch using a constraint-solving approach. For example, if the user's NL explanation mentions "not too close to the sidewalk", the physical demonstrations are needed to understand what the user considers "too close". For this reason, `SYNAPSE-Learn` utilizes a separate parameter synthesis procedure to determine suitable numeric parameters from the physical demonstrations.

The remainder of this section explains these aspects of the learning procedure in more detail.

1) Concept Library Update.: SYNAPSE analyzes the user's natural language explanation e to extract concepts of interest. We differentiate between two types of concepts: (1) entities (e.g., car, door, sidewalk) and (2) predicates (e.g., far, near). Because SYNAPSE uses an open-vocabulary visual language model to find entities of interest in the current observation, new entity concepts do not require interacting with the user. On the other hand, if the natural language explanation contains new predicates that are not part of the existing concept library, SYNAPSE needs to query the user to provide suitable demonstrations.

First, the `ExtractEntities` procedure grounds the entities used in the NL description and cross-references them against existing entities in the concept library. Any new entities are added to the concept library without requiring user interaction, as we assume that any entity can be extracted from the observation using a modern VLM. Next, we must extract new *predicates* from the natural language description. Since the semantics of these predicates are not known a priori (unless they are already in the concept library), we must query the user to learn their semantics. Thus, the `QueryUserForDemonstration` procedure obtains new demonstrations, which are then used to synthe-

TABLE I: Mean IOU (%) results for the three concepts. The train set represents the training set for neural baselines – SYNAPSE only needs 29 demonstrations (from the train area on UT Campus), and the visual-language baselines have not been fine-tuned.

	CONTINGENCY			DROPOFF			PARKING	
	train	in-test	out-test	train	in-test	out-test	train	out-test
Synapse	**77.64**	**76.29**	**74.07**	**79.32**	**80.18**	**80.72**	68.60	**62.75**
SF-RGB-b0	70.49	62.75	57.42	72.99	68.13	52.21	57.68	49.66
SF-RGB-b5	74.59	70.48	56.00	77.26	72.83	55.04	68.63	52.90
SF-RGBD-b0	72.17	67.23	54.75	74.33	69.80	54.90	60.90	50.69
SF-RGBD-b5	76.48	67.81	56.11	77.69	70.70	52.39	**71.06**	49.99
GPT4V	26.49	29.64	30.40	30.49	32.96	37.15	38.41	40.18
GPT4V+	28.80	29.00	33.84	38.87	38.13	39.31	41.37	39.77
VisProg	45.62	45.63	44.98	46.29	47.49	48.07	39.42	40.99
VisProg+	38.94	39.21	41.83	39.17	39.44	43.14	38.87	38.99

size the implementation of the new predicate through recursive invocation of SYNAPSE-Learn. Thus, when the UpdateConceptLibrary procedure terminates, the new concept library $\mathscr{C}'$ contains all entities and predicates of interest.

2) Program Sketch Synthesis.: Once SYNAPSE has all the required concepts as part of its library, it uses a large language model to synthesize a suitable program sketch based on the natural language description and concept library. In particular, it first prompts the LLM to translate the NL explanation e to a pair (Φ, p) where Φ is a formula (in conjunctive normal form) over the predicates in the concept library and p is the user's preference. Then, in a second step, SYNAPSE prompts the LLM to update the previous sketch $\hat{\pi}$ to a new one $\hat{\pi}'$ such that $\hat{\pi}'$ returns p when Φ evaluates to True. We found this two-stage process of first converting the NL explanation to a CNF formula and then prompting the LLM to repair the old sketch to work better in practice compared to prompting the LLM directly with all inputs.

III. EVALUATION

To test the effectiveness of SYNAPSE, we evaluate it on three mobility-related preferential concepts relevant to mobile robotics and autonomous driving domains: a) CONTINGENCY: *What is a good spot for a robot to pull over to in case of an emergency?*, b) DROPOFF: *What is a good location for an autonomous taxi to stop and drop-off a customer?*, and c) PARKING: *What is a good location for parking an autonomous car?*.

To evaluate SYNAPSE, we create a dataset of 815 labeled images collected on UT Campus, where the labels mark the locations on the images that are consistent with the intended user preference for each of the three tasks. We split the dataset into three sets: train, in-distribution test, and out-of-distribution test sets. The train and in-distribution sets belong to the same part of the UT Campus, while the out-of-distribution set belongs to a different area. Table I shows the comparison against various baselines. We use mean Intersection-Over-Union (mIOU) as the metric and evaluate the following baselines: (1) pure neural models based on SegFormer [37] architecture with pretrained weights, with and without depth input, fine-tuned on our custom dataset; (2) GPT4 [38] with vision capabilities, and (3) VisProg [29]. The '+' for the latter two indicate additional prompting that provides additional information about the underlying reasoning behind the preferential concept.

We find that SYNAPSE outperforms all baselines, and improves on the closest baseline by a significant margin on out-of-distribution test data — 74.07 vs. 57.42 for CONTINGENCY, 80.72 vs. 55.04 for DROPOFF, and 62.75 vs. 52.90 for PARKING. Further, even though SYNAPSE is trained on an order of magnitude fewer samples (for instance, 29 demonstration for CONTINGENCY) than neural baselines (for instance, 224 images for CONTINGENCY), it performs at-par, if not better, on the train dataset.

IV. CONCLUSION, LIMITATIONS, & FUTURE WORKS

We presented SYNAPSE, a data-efficient, neuro-symbolic framework for learning preferential concepts from a small number of human demonstrations. The framework utilises a novel combination of visual parsing, large language models, and program synthesis to represent preferences as interpretable programs that can be synthesized from demonstrations. We experimentally showed that SYNAPSE achieves strong generalization on new data and that it outperforms the baselines by a large margin ($\approx 15\%$ mIOU).

This work has three potential limitations that could be addressed in future research. First, SYNAPSE relies substantially on how good the underlying neural modules are and their capabilities, specifically, the zero-shot LLMs and VLMs. In our experiments, we observe that a careful selection of parameters and verbose prompting is needed to achieve best performance. Second, SYNAPSE relies on the quality of the user's physical demonstrations for accurate parameter synthesis. In practice, the demonstrations may be noisy and imperfect, and SYNAPSE tries to compensate for slight inconsistencies in the user demonstration by using a constrained optimization approach based on MaxSMT. Third, another limitation of operating on real-world data with SYNAPSE is the incomplete depth information – SYNAPSE approximates depth at all locations by interpolation, however, that introduces noise into the quantitative evaluation. Better approaches for scene completion [39] would reduce such noise.

References

[1] J. Andreas, M. Rohrbach, T. Darrell, and D. Klein, "Neural module networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 39–48.

[2] S. Srivastava, I. Labutov, and T. Mitchell, "Joint concept learning and semantic parsing from natural language explanations," in *Proceedings of the 2017 conference on empirical methods in natural language processing*, 2017, pp. 1527–1536.

[3] J. Mao, C. Gan, P. Kohli, J. B. Tenenbaum, and J. Wu, "The neuro-symbolic concept learner: Interpreting scenes, words, and sentences from natural supervision," *arXiv preprint arXiv:1904.12584*, 2019.

[4] A. Murali, A. Sehgal, P. Krogmeier, and P. Madhusudan, "Composing neural learning and symbolic reasoning with an application to visual discrimination," *arXiv preprint arXiv:1907.05878*, 2019.

[5] T. J.-J. Li, M. Radensky, J. Jia, K. Singarajah, T. M. Mitchell, and B. A. Myers, "Interactive task and concept learning from natural language instructions and gui demonstrations," *arXiv preprint arXiv:1909.00031*, 2019.

[6] C. Han, J. Mao, C. Gan, J. Tenenbaum, and J. Wu, "Visual concept-metaconcept learning," *Advances in Neural Information Processing Systems*, vol. 32, 2019.

[7] P. W. Koh, T. Nguyen, Y. S. Tang, S. Mussmann, E. Pierson, B. Kim, and P. Liang, "Concept bottleneck models," in *International conference on machine learning*. PMLR, 2020, pp. 5338–5348.

[8] A. Xie and C. Finn, "Lifelong robotic reinforcement learning by retaining experiences," in *Conference on Lifelong Learning Agents*. PMLR, 2022, pp. 838–855.

[9] Z. Chen, J. Mao, J. Wu, K.-Y. K. Wong, J. B. Tenenbaum, and C. Gan, "Grounding physical concepts of objects and events through dynamic visual reasoning," *arXiv preprint arXiv:2103.16564*, 2021.

[10] L. Mei, J. Mao, Z. Wang, C. Gan, and J. B. Tenenbaum, "Falcon: fast visual concept learning by integrating images, linguistic descriptions, and conceptual relations," *arXiv preprint arXiv:2203.16639*, 2022.

[11] K. Namasivayam, H. Singh, V. Bindal, A. Tuli, V. Agrawal, R. Jain, P. Singla, and R. Paul, "Learning neuro-symbolic programs for language guided robot manipulation," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2023, pp. 7973–7980.

[12] B. Kane, F. Gervits, M. Scheutz, and M. Marge, "A system for robot concept learning through situated dialogue," in *Proceedings of the 23rd Annual Meeting of the Special Interest Group on Discourse and Dialogue*, 2022, pp. 659–662.

[13] T. Silver, A. Athalye, J. B. Tenenbaum, T. Lozano-Perez, and L. P. Kaelbling, "Learning neuro-symbolic skills for bilevel planning," *arXiv preprint arXiv:2206.10680*, 2022.

[14] J. Hsu, J. Mao, and J. Wu, "Ns3d: Neuro-symbolic grounding of 3d objects and relations," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 2614–2623.

[15] G. Wang, Y. Xie, Y. Jiang, A. Mandlekar, C. Xiao, Y. Zhu, L. Fan, and A. Anandkumar, "Voyager: An open-ended embodied agent with large language models," *arXiv preprint arXiv:2305.16291*, 2023.

[16] J. Hsu, J. Mao, J. B. Tenenbaum, and J. Wu, "What's left? concept grounding with logic-enhanced foundation models," *arXiv preprint arXiv:2310.16035*, 2023.

[17] J. Johnson, B. Hariharan, L. Van Der Maaten, L. Fei-Fei, C. Lawrence Zitnick, and R. Girshick, "Clevr: A diagnostic dataset for compositional language and elementary visual reasoning," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 2901–2910.

[18] A. Wilson, A. Fern, and P. Tadepalli, "A bayesian approach for policy learning from trajectory preference queries," *Advances in neural information processing systems*, vol. 25, 2012.

[19] R. Busa-Fekete, B. Szörényi, P. Weng, W. Cheng, and E. Hüllermeier, "Preference-based evolutionary direct policy search," in *ICRA Workshop on autonomous learning*, vol. 2, 2013.

[20] C. Wirth, J. Fürnkranz, and G. Neumann, "Model-free preference-based reinforcement learning," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 30, no. 1, 2016.

[21] R. Akrour, M. Schoenauer, M. Sebag, and J.-C. Souplet, "Programming by feedback," in *International Conference on Machine Learning*, no. 32. JMLR. org, 2014, pp. 1503–1511.

[22] P. F. Christiano, J. Leike, T. Brown, M. Martic, S. Legg, and D. Amodei, "Deep reinforcement learning from human preferences," *Advances in neural information processing systems*, vol. 30, 2017.

[23] D. Sadigh, A. D. Dragan, S. Sastry, and S. A. Seshia, *Active preference-based learning of reward functions*, 2017.

[24] A. Bestick, R. Pandya, R. Bajcsy, and A. D. Dragan, "Learning human ergonomic preferences for handovers," in *2018 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2018, pp. 3257–3264.

[25] Y. Cui and S. Niekum, "Active reward learning from critiques," in *2018 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2018, pp. 6907–6914.

[26] R. Zhang, D. Bansal, Y. Hao, A. Hiranaka, J. Gao, C. Wang, R. Martín-Martín, L. Fei-Fei, and J. Wu, "A dual representation framework for robot learning with human guidance," in *Conference on Robot Learning*. PMLR, 2023, pp. 738–750.

[27] J. Fürnkranz, E. Hüllermeier, W. Cheng, and S.-H. Park, "Preference-based reinforcement learning: a formal framework and a policy iteration algorithm," *Machine learning*, vol. 89, pp. 123–156, 2012.

[28] Z. Hu, F. Lucchetti, C. Schlesinger, Y. Saxena, A. Freeman, S. Modak, A. Guha, and J. Biswas, "Deploying and evaluating llms to program service mobile robots," *arXiv preprint arXiv:2311.11183*, 2023.

[29] T. Gupta and A. Kembhavi, "Visual programming: Compositional visual reasoning without training," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 14 953–14 962.

[30] D. Surís, S. Menon, and C. Vondrick, "Vipergpt: Visual inference via python execution for reasoning," *arXiv preprint arXiv:2303.08128*, 2023.

[31] N. Wake, A. Kanehira, K. Sasabuchi, J. Takamatsu, and K. Ikeuchi, "Gpt-4v (ision) for robotics: Multimodal task planning from human demonstration," *arXiv preprint arXiv:2311.12015*, 2023.

[32] I. Singh, V. Blukis, A. Mousavian, A. Goyal, D. Xu, J. Tremblay, D. Fox, J. Thomason, and A. Garg, "Progprompt: Generating situated robot task plans using large language models," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2023, pp. 11 523–11 530.

[33] Y. Ding, X. Zhang, C. Paxton, and S. Zhang, "Task and motion planning with large language models for object rearrangement," *arXiv preprint arXiv:2303.06247*, 2023.

[34] B. Liu, Y. Jiang, X. Zhang, Q. Liu, S. Zhang, J. Biswas, and P. Stone, "Llm+ p: Empowering large language models with optimal planning proficiency," *arXiv preprint arXiv:2304.11477*, 2023.

[35] R. Wang, J. Mao, J. Hsu, H. Zhao, J. Wu, and Y. Gao, "Programmatically grounded, compositionally generalizable robotic manipulation," *arXiv preprint arXiv:2304.13826*, 2023.

[36] J. Holtz, A. Guha, and J. Biswas, "Robot action selection learning via layered dimension informed program synthesis," in *Conference on Robot Learning*. PMLR, 2021, pp. 1471–1480.

[37] E. Xie, W. Wang, Z. Yu, A. Anandkumar, J. M. Alvarez, and P. Luo, "Segformer: Simple and efficient design for semantic segmentation with transformers," *Advances in Neural Information Processing Systems*, vol. 34, pp. 12 077–12 090, 2021.

[38] J. Achiam, S. Adler, S. Agarwal, L. Ahmad, I. Akkaya, F. L. Aleman, D. Almeida, J. Altenschmidt, S. Altman, S. Anadkat, *et al.*, "Gpt-4 technical report," *arXiv preprint arXiv:2303.08774*, 2023.

[39] X. Meng, N. Hatch, A. Lambert, A. Li, N. Wagener, M. Schmittle, J. Lee, W. Yuan, Z. Chen, S. Deng, *et al.*, "Terrainnet: Visual modeling of complex terrain for high-speed, off-road navigation," *arXiv preprint arXiv:2303.15771*, 2023.