A Unifying Perspective on Language-Based Task Representations for Robot Control

Maximilian Tölle[1,*] **Boris Belousov**[1] **Jan Peters**[1,2,3,4]

[1]DFKI GmbH, SAIROL, Germany [2]TU Darmstadt, IAS, Germany
[3]Hessian.AI, Germany [4]Centre for Cognitive Science, Germany

Abstract: Natural language is becoming increasingly important in robot control for both high-level planning and goal-directed conditioning of motor skills. While a number of solutions have been proposed already, it is yet to be seen what architecture will succeed in seamlessly integrating language, vision, and action. To better understand the landscape of existing methods, we propose to view the algorithms from the perspective of "Language-Based Task Representations", i.e., categorizing the methods that condition robot action generation on natural language commands according to their task representation and embedding architecture. Our proposed taxonomy intuitively groups existing algorithms, highlights their commonalities and distinctions, and suggests directions for further investigation.

Keywords: Language-Based Task Representations, Robot Control

1 Introduction

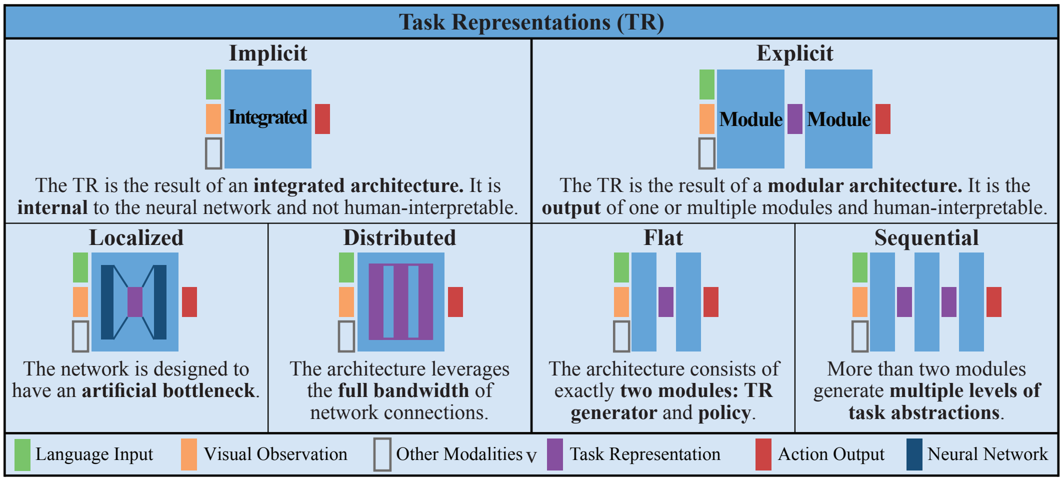
The rapid advent of Large Language Models (LLM) and related multi-modal architectures has demonstrated the power of general-purpose self-supervised pre-training on multi-task datasets [1]. These advancements have been quickly adopted in robotics for task planning, reasoning, skill conditioning, and more [2, 3, 4]. However, the search for the most general and efficient system architecture that integrates language representations with vision and other sensory modalities as well as robot actions is still ongoing. In this paper, we systematically investigate the landscape of the methods in this area, published predominantly within the last four years, that incorporate natural language representations into robot control. Due to the fast pace of the field, we may have not covered all the papers, but we made the best effort to highlight the representative papers in each category of methods.

As a unifying perspective, we propose to consider **language-based task representations**. The key question for connecting language to action is *How should language be grounded in observations such that a downstream policy can generate task-specific actions for fine-grained control?* This overall problem can be split into two parts. First, learning 'sufficient' task representations that contain all task-relevant information. Second, training a task-conditioned policy that utilizes these learned representations. In the following, we show how different approaches solve these two subproblems, and we identify and contrast their corresponding design choices.

This paper focuses on the interface between language commands and low-level robot control actions, i.e., **language-conditioned robot control**. For applications of language to high-level planning/decision making and embodied AI, we refer to [5, 6]. We further restrict the scope to only those approaches that have been demonstrated on real or simulated robot platforms. In addition, all considered algorithms are vision-based and output discrete or continuous control actions that can modulate the robot behavior at a fine-grained level, such as end-effector waypoints. Thus, we leave out the approaches that output language skill labels that are used for calling pre-defined skills from an existing skill-library. Instead, we are interested in the interplay between the representations of language, vision, and action.

[*] Correspondence: maximilian.toelle@dfki.de

2nd Workshop on Language and Robot Learning, Conference on Robot Learning (CoRL 2023), Atlanta, USA.



The TR is the result of an **integrated architecture.** It is **internal** to the neural network and not human-interpretable.

The TR is the result of a **modular architecture.** It is the **output** of one or multiple modules and human-interpretable.

The network is designed to have an **artificial bottleneck**.

The architecture leverages the **full bandwidth** of network connections.

The architecture consists of exactly **two modules: TR generator** and **policy**.

More than two modules generate **multiple levels of task abstractions**.

Figure 1: Proposed taxonomy of task representations. The two main categories of the taxonomy are **Implicit** and **Explicit**, each containing two sub-categories. Implicit Representations can be **Localized** or **Distributed**. Explicit Representations are either **Flat** or **Sequential**.

2 Language-Based Task Representations for Robot Control

A **task** refers to a specific goal or objective described in natural language that a robot needs to achieve in a given environment. It can be formulated as a process, such as *grab the block*, or as a description of the target outcome, e.g., *red block on blue block*. A task itself is abstract and context-independent, since the same task can be given to different robots in different environments. Only the task execution is context-dependent.

A **Task Tepresentation (TR)** encodes all the necessary information about the task in the context of the current observation or observation history regarding *what* to achieve, such that the downstream task-conditioned policy can figure out *how* to achieve it. Therefore, if task representations are **disentangled**, the policy can easily distinguish between and generalize across tasks.

Task representations can have a range of implementations, from a raw concatenation of representations of each individual modality, e.g., vision and language, to a refined and processed output, such as a list of end-effector waypoints (see Fig. 1). The choice of the task representation is intertwined with the choice of the policy implementation, i.e., if the task representation is **unstructured** (e.g., concatenation of embeddings), the policy needs to learn a complex mapping to generate actions, whereas a very **structured** task representation, (e.g., an action-value function) allows the policy to be extremely simple (e.g., a maximization operator). Figure 1 shows our taxonomy of task representations, which has two levels: implicit vs. explicit, and localized vs. distributed. The following sections describe each category in greater detail.

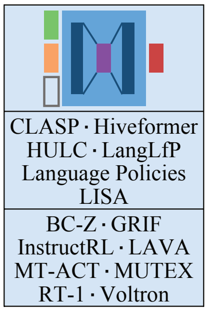
2.1 Implicit Task Representations

An **Implicit Task Representation (ITR)** is internal to the neural network and is not directly human-interpretable, such as an embedding vector or weights of a whole neural network. ITR's are commonly produced by auto-encoding architectures or via self-attention/cross-attention modules. There is a number of distinguishing features of different implicit task representations. On the highest level, a categorization into **localized** and **distributed** representations can be made.

2.1.1 Implicit Localized Task Representations

An **Implicit Localized Task Representation (ILTR)** captures all task-relevant information within a single concrete abstraction, e.g., a latent vector (see Fig. 2). An ILTR can be seen as **grounding** language instructions in the visual observations of the robot. Given the ILTR, robot actions can be generated in a context-specific and task-directed manner by conditioning the policy on the ILTR.

A key question here is *What information does the ILTR encode and how much does the policy need to learn?* Therefore, it is crucial to consider how the **policy conditioning** mechanism is implemented: whether the policy gets raw observations as input in addition to the task representation, or whether it only gets the task representation.



CLASP · Hiveformer
HULC · LangLfP
Language Policies
LISA

BC-Z · GRIF
InstructRL · LAVA
MT-ACT · MUTEX
RT-1 · Voltron

Figure 2: Implicit Localized Task Representations (ILTR).

Observation-Conditioned Policies

A large set of methods use task representations as a conditioning/context variable $\boldsymbol{c}_t$, i.e., as an additional input to the policy, $\boldsymbol{\pi}(\boldsymbol{a}_t|\boldsymbol{o}_t, \boldsymbol{c}_t(\boldsymbol{o}_{0:t}, \boldsymbol{l}))$, along with the current observation $\boldsymbol{o}_t$. The **task representation** $\boldsymbol{c}_t$ is conditioned on a **language embedding** $\boldsymbol{l}$ and the **observation history** $\boldsymbol{o}_{0:t}$, thus encoding a skill to execute or a goal to reach. Still, the policy gets the current observation $\boldsymbol{o}_t$ as an additional direct input. Note that the history-dependence may be incorporated to a varying degree, distinguishing between **static context**, $\boldsymbol{c}_t(\boldsymbol{o}_0, \boldsymbol{l})$, used in [7, 8, 9], and **dynamic context**, $\boldsymbol{c}_t(\boldsymbol{o}_{0:t}, \boldsymbol{l})$, employed in [10, 11, 12]. Below we describe the algorithms that employ observation-conditioned policies in more detail, paying particular attention to feature representation. A list overview of these methods can be found in the top row of Fig. 2.

There is a significant variability among the methods in how they implement **feature alignment**, i.e., the combination of vision, language, proprioception, and other modalities to obtain a task representation. First, we describe methods that use static context embeddings, i.e., the task representation does not change during the task execution. *LangLfP* [7] and *HULC* [8] employ generic **latent representations**, given as the latent distribution of a Sequence-to-Sequence Conditional Variational Autoencoder (seq2seq cVAE) [13] tailored to robotic action generation. In this case, the ILTR is a sampled **latent plan** that encodes how to get from the current state to a latent goal state. *CLASP* [9] focuses on learning a **shared representation** for a given language task and its corresponding state-action trajectory. Separate encoders of both modalities output the parameters of a Gaussian distribution. To align the modalities, CLASP leverages the contrastive loss of CLIP [14]. Additional loss components are introduced through auxiliary learning tasks. Within behavior generation, the authors generate 2D delta actions for tabletop rearrangement based on the current observation and the language embedding sampled from the shared distributional representation.

Observation-conditioned policies with dynamic context employ further feature encoding architectures. *LanguagePolicies* [10] leverage **object-centric representations**, aligning features of detected objects in the current observation with a given language task via an attention network. The output gets concatenated with the language task encoding and passed through a single fully connected layer to yield an ILTR, on which an underlying movement primitive is conditioned. *Hiveformer* [11] additionally incorporates multi-view observations and the full observation history into the task representation. They employ the default CLIP model for language task embedding, while training a UNet [15] for multi-view observation encoding. A **multimodal transformer**, that relates the encoded current observation with the encoded observation history and language task embedding, outputs the ILTR on which the next end-effector pose and gripper state gets predicted. Along with continuous embeddings, **discrete task representations** can be utilized. *LISA* [12] leverages VQ-VAE [16] to learn a low-dimensional discrete codebook of skills, similar to HULC. Given a language-labeled state-action trajectory, a causal transformer predicts a skill code which gets mapped to the closest vector in the codebook; this codebook vector plays the role of the discrete ILTR in this case.

Task-Conditioned Policies

The second big category of algorithms conditions the policy $\boldsymbol{\pi}$ on the task representation $\boldsymbol{c}_t$ alone, $\boldsymbol{\pi}(\boldsymbol{a}_t|\boldsymbol{c}_t(\boldsymbol{o}_{0:t}, \boldsymbol{l}))$, meaning that the current observation $\boldsymbol{o}_t$ gets completely integrated into the context variable $\boldsymbol{c}_t$ together with the history $\boldsymbol{o}_{0:t-1}$ and the language embedding $\boldsymbol{l}$ before entering the policy. Therefore, all task representations in this category are **dynamic**, i.e., the context variable $\boldsymbol{c}_t$ depends on the current observation $\boldsymbol{o}_t$ in a non-trivial manner. An overview list of methods in this category can be found in the bottow row of Fig. 2.

BC-Z [17] and *GRIF* [18] specify a task using either language or vision and map both modalities into a shared task embedding space, similar to LangLfP and HULC. BC-Z leverages human demonstration videos next to language tasks, while GRIF uses pairs of (initial, goal)-images. In addition, GRIF explicitly aligns both task specifications by contrastive learning before training the downstream policy, while BC-Z trains the task encoders and downstream policy together. Both methods obtain the ILTR by using **Feature-wise Linear Modulation (FiLM) layers** [19] to condition the ResNet encoding of the current visual observation on an obtained latent task embedding. The ouput passes fully connected layer to predict deterministic continuous robotic actions in BC-Z and stochastic continous actions in GRIF.

Instead of using the current observation alone, *RT-1* [20] encodes an **observation history** of 6 images using EfficientNet [21] and conditions the architecture on language tasks using FiLM-layers. Obtained vision-language tokens are further compressed by a TokenLearner [22] whose output corresponds to the ILTR. In the following, a decoder-only transformer policy generates action tokens for an 11-DoF discretized action space (6-DoF end-effector pose + gripper state + 3-DoF base + action mode). The FiLM layers are further used in *MT-ACT* [23]. Similar to LangLfP and HULC, they leverage a cVAE architecture to capture the **multimodal trajectory distribution** in the teleoperation dataset. A distinguishing feature is the combined use of FiLM-layers and encoder-decoder Transformer. MT-ACT produces a sequence of 8-DoF robot actions (7-DoF joint positions + gripper state) where the next action is the temporal ensemble of predicted actions for the next timestep[24].

Next to FiLM-conditioning, some methods also leverage the attention mechanism for vision-language alignment. *LAVA* [3] fuses image encodings with language embeddings via **cross-attention** frame-by-frame, and subsequently applies self-attention to the obtained vision-language embedding sequence, average-pooling the resulting output time series over the time dimension to obtain the ILTR. The downstream policy is a deep residual MLP which controls the robot end-effector with 2D delta actions in a tabletop rearrangement scenario. Instead of separately encoding the vision and language modality, *InstructRL* [25] leverages a pre-trained **Multimodal Masked Autoencoder (M3AE)** [26] to encode both modalities together. The ILTR consists of the limited history of vision-language encodings of all cameras as well as linear mappings of proprioception and action information to the same representation space. InstructRL's transformer-based policy captures the relationships among all the different features and predicts action tokens which are then passed into a feature map to predict the next 7-DoF keyframe action (6-DoF end-effector pose + gripper state). A keyframe action fulfills at least one of two criterions: the gripper state changes or velocities approach near zero [27]. *Voltron* [28] also jointly encodes the vision and language modality by following a masked autoencoding pipeline. However, while M3AE learns to purely reconstruct masked (image, text) pairs, Voltron introduces a trade-off parameter between language-conditioned image reconstruction and **image-based language generation**. The choice of language generation over masked language modeling is especially beneficial for short, predictable language labels. The learned encoder weights get frozen and the latent representation, corresponding to the ILTR, is used for downstream language-conditioned robotic manipulation by MLP-mapping to the next best end-effector keyframe pose.

MUTEX [29] combines multiple features of previously described approaches. They specify the task in six different modalities, from video demonstrations over text instructions to speech goals. Task specifications are separately encoded, masked and projected into a common embedding space. Resulting tokens are fused with observation tokens through multiple self- and cross-attention layer before being passed to a Transformer decoder to predict a Gaussian mixture model for continuous actions and the masked tokens.

2.1.2 Implicit Distributed Task Representations

An **Implicit Distributed Task Representation (IDTR)**, in contrast to the localized TR, cannot be pinpointed to one particular location within the neural network but is rather spread across the weights and layers of abstraction. The models in this category are potentially very powerful because they do not have the artificial bottlenecks of ILTR's with pre-defined dimensionality but rather can leverage the full bandwidth of network connections. The drawback, however, is that the ease of sharing of the embeddings and the modularity are reduced in these models.

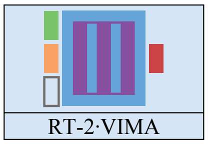


Figure 3: Implicit Distributed Task Representations (IDTR).

The two notable IDTR models are *VIMA* [30] and *RT-2* [31]. VIMA leverages **object-centric representations**, similar to Language Policies [10]. They encode an observation-action history conditioned on a given multimodal prompt through a series of cross-attention layers alternating with self-attention layers, thereby intermixing the representations at multiple levels in the hierarchy. The output is an action token that gets mapped to start and goal SE(2) poses. Due to its architecture, VIMA allows for **interleaving text and images within the task specification**. RT-2, on the other hand, co-fine-tunes large vision-language models [32, 4] to directy perform closed-loop robot control by **representing robot actions as language tokens**. Each dimension of the 7-DoF action space (6-DoF end-effector pose + gripper extension) is uniformly discretized into 256 bins denoted by successive natural numbers corresponding to tokens. Thus, RT-2 represents a truly end-to-end monolithic vision-language-*action* model capable of online robot control at 1-5Hz depending on the model size.

2.2 Explicit Task Representations

An **Explicit Task Representation (ETR)** is given by a structured and generally human-interpretable abstraction, such as a reward function or a value function. ETR's enable modular design, with outputs of each module being testable independently.

2.2.1 Explicit Flat Task Representations

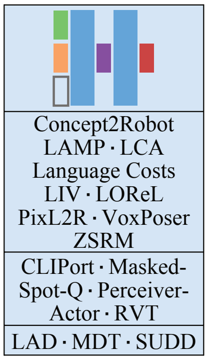


Figure 4: Explicit Flat Task Representations (EFTR).

An **Explicit Flat Task Representation (EFTR)** is an ETR that is given by a concrete localized object at the output of a neural network, e.g., a value map. The EFTR then serves as a compact representation of all the information needed for the policy to make the decision what action to take. We discern *Reward*-, *Value*-, and *Score*-based EFTR's (R-, V-, and S-EFTR's, respectively). Similar to ILTR, the distinction between these categories is in the amount of computation and reasoning the policy needs to do, e.g., for R-EFTR's, the policy needs to optimize the long-term sum of rewards, whereas V-EFTR directly provides the Q-function which only needs to be maximized.

Reward-Based Task Representations

Reward-Based Explicit Flat Task Representations (R-EFTR) provide a sparse or dense supervision signal to the policy in the form of a reward/cost. The policy is then either trained to optimize this reward or a trajectory planner is used. In the following, we describe the algorithms in this category, paying special attention to how the rewards are generated and how the subsequent policy is implemented and trained.

Reward Generation. A common approach to reward generation is to **train a reward model from scratch**. *PixL2R* [33] and *Concept2Robot* [34] both predict whether a given video sequence and a language task are related. While *Concept2Robot* trains a video classification network to evaluate the video of a fully recorded trajectory, *PixL2R* uses a regression loss to also predict the relatedness score of partial trajectories. The binary classifier of *LOReL* [35] instead predicts whether going from an initial to the current observation solves the given language task.

Rewards can also be generated by **pre-trained foundation models**. *ZSRM* [36] and *LCA* [37] fine-tune CLIP [14] on in-domain data and leverage the cosine similarity between the CLIP-encoded task description and the image observation as the reward. *LAMP* [38] chooses pre-trained R3M [39] over other vision-language models as it is explicitly trained to understand temporal information within videos. LAMP leverages the score predictor of R3M to measure task progress under a given language task and weights this reward with a novelty-based exploration score. *VoxPoser* [40] generates

voxel value maps in 3D observation space using the reasoning and coding capabilities of an LLM. Based on a given instruction, the LLM calls perception APIs to obtain spatial-geometric information of the scene and then generates numpy operations to manipulate 3D reward arrays. This results in sparse voxel value maps which get densified via smoothing operations.

Alternatively, **preferences** or **value differences** can be utilized for reward generation. *LIV* [41] learns an embedding space where the similarity metric between an observation and a language task corresponds to an **implicit value function**. The value difference of two subsequent states is then used as a reward. *Language Costs* [42] uses language preferences to correct hand-crafted costmaps. A generative model is trained to output language-conditioned 2D costmaps and binary masks over them which are combined using element-wise multiplication. The result gets added to the hand-crafted costmaps to close the misalignment gap between the intended and executed tasks.

Policy Learning. Depending on how the learned reward model is used for policy learning, we distinguish between pure **learned-reward** methods that use only the learned rewards and **composite-reward** methods that combine the learned and the environment rewards. We further distinguish between **model-free** and **model-based** methods according to the RL algorithm used, noting that the learned reward model can also be directly used for Model Predictive Control (MPC) [35, 41, 42, 40].

Concept2Robot [34] and *ZSRM* [36] are pure learned-reward methods, that employ model-free RL to train single-task policies which are subsequently used to train a multi-task policy via **Behavior Cloning (BC)**. This two-step process is more stable for **multi-task** training as the learned reward signal may be noisy and biased towards some tasks. *LAMP* [38] is another pure learned-reward method that similarly questions the reliability of the vision-language reward models and suggests using them only for **pre-training**. LAMP pre-trains a model-based RL algorithm using solely the learned reward model and fine-tunes pre-trained skills using downstream environment reward. In comparison to a randomly initialized agent, the pre-trained agent is biased to explore semantic meaningful paths and is able to quickly adapt to unseen task rewards during fine-tuning.

On the other hand, *PixL2R* [33] is a composite-reward method, that uses the predicted relatedness score as an intermediate reward for **potential-based reward shaping** [43] to train single-task model-free RL policies. Likewise, as a composite-reward method, *LCA* [37] leverages a learned reward model to provide additional guidance to a model-free RL agent by generating and evaluating sub-tasks on which a **self-imitation** policy is trained in a collect-infer cycle [44].

Value-Based Task Representations

While reward-based representations still require the dynamics model of the environment or need to train a policy to optimize the reward, **Value-Based Explicit Flat Task Representations (V-EFTR)** bake the dynamics together with the reward into a Q-function. By taking the maximum argument over the Q-function, the next optimal end-effector position or pose is obtained which can be reached using a motion planner or movement primitive.

Masked-SPOT-Q [45] represents language-conditioned pick-and-place locations by **pixel-based Q-functions**. It leverages the vision-only robot manipulation model SPOT-Q [46] to determine *how to act*, developing a transformer model that predicts *where to act*. This transformer model maps a language task and a visual observation into distinct spatial image masks for picking and placing. These masks are combined with the Q-function of SPOT-Q to obtain language-conditioned pixel-wise Q-values.

In contrast, *CLIPort* [47] not only predicts pick-and-place locations but full $SE(2)$ end-effector poses. It extends Transporter [48] to enable language-conditioning by adding a **semantic network** to the spatial Transporter network. The semantic network consists of the image and language encoders of pre-trained CLIP [14] and an introduced decoder which outputs a Q-function-based affordance map used for pick-and-place end-effector pose prediction.

One step further, *Perceiver-Actor* [49] extends the formulation of Q-functions as task representations to manipulation tasks in SE(3). Following [50], Perceiver-Actor employs **3D voxelization** to build a structured observation space, and utilizes 6-DoF end-effector keyframes (waypoints) as actions.

To deal with long vision-language sequences, *Perceiver-Transformer* [51] maps them to a lower-dimensional latent space before processing and decoding the embeddings into separate action-value functions for the end-effector's translation, rotation and binary gripper state as well as a binary collide variable of the used motion planner.

The voxel-based representations, however, significantly increase the training time. Therefore, *Robotic View Transformer (RVT)* [52] proposes an image-based method for 3D object manipulation, that instead of voxelizing multi-view observations, reconstructs a **point cloud** of the scene and re-renders it from several virtual viewpoints before producing image tokens. The image tokens are fed into a transformer model together with the encoded language task and gripper state. The output of the transformer gets decoded into Q-functions as in Perceiver-Actor.

Score-Based Task Representations

Next to predicting the Q-function as the task representation, an increasingly popular approach is to model the **score function** $\nabla_{\boldsymbol{a}} Q(\boldsymbol{s}, \boldsymbol{a})$. *SUDD* [53], which adds language-conditioning to Diffusion Policy [54], *LAD* [55] and *MDT* [56] are all formulated as **Denoising Diffusion Probabilistic Models (DDPMs)** [57, 58, 59]. They train a noise prediction network to model the score function of the conditional action distribution. During inference, they use the generative process of Denoising Diffusion Implicit Models (DDIM) [60] to optimize a Gaussian noise sample with respect to the gradient field to obtain a 7-DoF (6-Dof end-effector + gripper state) action sequence. The action plan gets executed open-loop until a new sequence is sampled. A distinguishing feature is the network architecture used for noise prediction. SUDD is based on Diffusion Policy which uses a decoder-only transformer while LAD uses a modified temporal U-Net [61], similar to latent diffusion models [62]. MDT allows for multimodal task specifications and uses an encoder-decoder transformer architecture to further tackle the challenge of modality alignment.

2.2.2 Explicit Sequential Task Representations

An **Explicit Sequential Task Representation (ESTR)** consists of multiple human-interpretable representations of a task, at different **levels of abstraction** or for different **subtasks**. The architecture is typically modular, with the output of each sub-module corresponding to a part of the overall task representation.

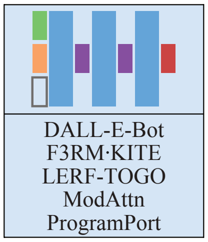


Figure 5: Explicit Sequential Task Representations (ESTR).

ESTR over **different levels of abstraction** is present in *DALL-E-Bot* [63], a modular approach built around pre-trained DALL-E 2 [64] for object rearrangement tasks. Here, object-level segmentation masks of an initial scene image and a DALL-E 2 **generated goal image** are being aligned to obtain the transformations between objects. These transformations are used for rearranging the objects with a sequence of pick-and-place operations. Each module involved in this process contributes a different level of abstraction to the ESTR, i.e. the generated rearrangement image, the segmentation masks, and the desired pick-and-place poses. Similarly, *KITE* [65] also uses different levels of abstraction for its task representation in a high-precision manipulation task. Given a fine-grained language task such as *pick up the stuffed bear by the ear*, the described object part gets represented by a **keypoint in the visual scene** and an LLM infers a task-matching language-labeled skill from a skill library. The selected skill consists of a learned policy that outputs **waypoints** and a **skill-specific controller** that outputs a low-level trajectory between the waypoints. The task representation is sequential because no single level of abstraction, i.e., the skill label, the keypoint, and the waypoints, is enough for precise low-level action generation.

Another line of research generates an ESTR around Neural Radiance Fields (NeRF) [66]. *F3RM* [67] generates a CLIP-based distilled feature field (DFF) and represents 6-DoF gripper demonstration poses in it. During a sequential inference process, they can infer gripper poses that are aligned with a given CLIP-encoded language task. *LERF-TOGO* [68] improves the spatial grouping of LERF [69] relevancy outputs. In their sequential process, they reconstruct the scene, render an object-centric point cloud, extract a 3D object mask and condition an instructed object part on this object mask to output a ranking of viable grasps.

Next to the different levels of task abstraction, the ESTR can also be composed of **different subtask representations**. *ModAttn* [70] divides a language task into subtasks where each subtask is assigned to a module inside a single overarching neural network. A module is defined by a supervised attention layer trained to maximize subtask-specific attention map elements provided by the user. **Modules are arranged in a sequence** where the output of one module corresponds to the input of the next one. Information can further be propagated through attention layers using slot attention [71]. The user can query each of the module outputs, e.g., the displacement of the end-effector from the object to manipulate. The output of all modules together corresponds to the ESTR. ModAttn outputs sequences of 6+1D gripper poses at every timestep. *ProgramPort* [72] is another approach that generates subtasks. It uses Combinatory Categorial Grammar [73] to parse a given language task into an **executable program of hierarchically structured operations**. An *operation* calls a functional module which consist of learnable neural networks for either visual grounding or action generation. E.g., *filter* calls a visual grounding module that localizes language-specified objects in the current observation and outputs a visual mask. The sequential output of the visual grounding modules corresponds to the ESTR. Based on the ESTR and the current observation, the action module *do* based on CLIPort outputs pick-and-place SE(2) end-effector poses.

3 Discussion & Conclusion

In this paper, we have proposed a classification of methods for language-based robot control from the perspective of **task representations**. We have identified several criteria which characterize different methods and allow for a systematic categorization, depending on whether the task representations are **explicit**, such as a value map over a voxelized 3D space, or **implicit**, such as a latent vector of an auto-encoding architecture. Within each category, we further grouped the representations into **localized** and **distributed**, according to the modularity of the corresponding network architectures.

Based on the presented overview, we can identify a number of commonalities and trends with regards to the types of representations and their uses in different scenarios. The grounding of **Implicit Localized Task Representations (ILTR)** in the visual observation space is commonly accomplished using auto-encoding architectures, self-/cross-attention layers, or via FiLM-conditioning (Sec 2.1.1). While early methods typically leveraged pre-trained encoder models from the vision and language domain, latest paper learn representations specifically tailored for downstream robotic action generation. A large number of approaches fall into the category of ILTR's thanks to the convenience of combining pre-trained encodings. However, the biggest and most powerful models, such as VIMA and RT-2, instead favor **Implicit Distributed Task Representations (IDTR)**, which leverage the full representational power of the neural networks, spreading the representation across the weights and layers. The trade off here is the loss of modularity and a higher computational cost (Sec. 2.1.2).

Explicit Flat Task Representations (EFTR) is the most widely used category of representations according to our survey (see Sec. 2.2.1), thanks to the generality and interpretability of the reward- and value-based task representations chiefly comprising this family of methods. Two important trends in this category are the use of discretized space representations such as 3D voxelization and the prediction of sparse keyframe actions. In addition, we note the rise in the popularity of score-based task representations, which build upon the success of diffusion-based models in image generation. Finally, **Explicit Sequential Task Representations (ESTR)** covered in Sec. 2.2.2, are yet relatively uncommon, presumably because they deal with hierarchies of abstractions and sub-tasks, which are still not well explored in the context of language-based robot control. Nevertheless, ESTR's are well suited for a more classical modular algorithm design paradigm, since every component produces an explicit and interpretable output and can be debugged separately.

There are still many open research questions, and we hope that our review provides a perspective and guidance within the vast design space of language-based robot control algorithms. In future work, we aim to further study the properties of task representations, focusing on transfer and generalization in the representation space, leveraging similarity and differences across tasks.

Acknowledgments

We acknowledge the grant “Einrichtung eines Labors des Deutschen Forschungszentrum für Künstliche Intelligenz (DFKI) an der Technischen Universität Darmstadt” of the Hessisches Ministerium für Wissenschaft und Kunst.

References

[1] R. Bommasani, D. A. Hudson, E. A. R. Altman, and S. Arora. On the Opportunities and Risks of Foundation Models. (arXiv:2108.07258), 2021.

[2] A. Zeng, M. Attarian, B. Ichter, K. Choromanski, A. Wong, S. Welker, F. Tombari, A. Purohit, M. S. Ryoo, V. Sindhwani, J. Lee, V. Vanhoucke, and P. Florence. Socratic Models: Composing Zero-Shot Multimodal Reasoning with Language. In *11th International Conference on Learning Representations (ICLR)*, 2023.

[3] C. Lynch, A. Wahid, J. Tompson, T. Ding, J. Betker, R. Baruch, T. Armstrong, and P. Florence. Interactive Language: Talking to Robots in Real Time. *IEEE Robotics and Automation Letters (RA-L)*, 2023.

[4] D. Driess, F. Xia, M. S. M. Sajjadi, C. Lynch, A. Chowdhery, B. Ichter, A. Wahid, J. Tompson, Q. Vuong, T. Yu, W. Huang, Y. Chebotar, P. Sermanet, D. Duckworth, S. Levine, V. Vanhoucke, K. Hausman, M. Toussaint, K. Greff, A. Zeng, I. Mordatch, and P. Florence. PaLM-E: An Embodied Multimodal Language Model. In *40th International Conference on Machine Learning (ICML)*, 2023.

[5] S. Tellex, N. Gopalan, H. Kress-Gazit, and C. Matuszek. Robots That Use Language. *Annual Review of Control, Robotics, and Autonomous Systems*, 3(1), 2020.

[6] M. Deitke, D. Batra, Y. Bisk, T. Campari, A. X. Chang, D. S. Chaplot, C. Chen, C. P. D’Arpino, K. Ehsani, A. Farhadi, L. Fei-Fei, A. Francis, C. Gan, K. Grauman, D. Hall, W. Han, U. Jain, A. Kembhavi, J. Krantz, S. Lee, C. Li, S. Majumder, O. Maksymets, R. Martín-Martín, R. Mottaghi, S. Raychaudhuri, M. Roberts, S. Savarese, M. Savva, M. Shridhar, N. Sünderhauf, A. Szot, B. Talbot, J. B. Tenenbaum, J. Thomason, A. Toshev, J. Truong, L. Weihs, and J. Wu. Retrospectives on the Embodied AI Workshop. (arXiv:2210.06849), 2022.

[7] C. Lynch* and P. Sermanet*. Language Conditioned Imitation Learning Over Unstructured Data. In *Robotics: Science and Systems (RSS)*, 2021.

[8] O. Mees, L. Hermann, and W. Burgard. What Matters in Language Conditioned Robotic Imitation Learning Over Unstructured Data. *IEEE Robotics and Automation Letters (RA-L)*, 7 (4), 2022.

[9] K. Rana, A. Melnik, and N. Sünderhauf. Contrastive Language, Action, and State Pre-training for Robot Learning. In *Pretraining for Robotics (PT4R) Workshop at International Conference on Robotics and Automation (ICRA)*, 2023.

[10] S. Stepputtis, J. Campbell, M. Phielipp, S. Lee, C. Baral, and H. B. Amor. Language-Conditioned Imitation Learning for Robot Manipulation Tasks. In *34th Conference on Neural Information Processing Systems (NeurIPS)*, 2020.

[11] P.-L. Guhur, S. Chen, R. Garcia, M. Tapaswi, I. Laptev, and C. Schmid. Instruction-driven history-aware policies for robotic manipulations. In *6th Conference on Robot Learning (CoRL)*, 2022.

[12] D. Garg, S. Vaidyanath, K. Kim, J. Song, and S. Ermon. LISA: Learning interpretable skill abstractions from language. In *36th Conference on Neural Information Processing Systems (NeurIPS)*, 2022.

[13] K. Sohn, H. Lee, and X. Yan. Learning Structured Output Representation using Deep Conditional Generative Models. In *29th Conference on Neural Information Processing Systems (NIPS)*, 2015.

[14] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, G. Krueger, and I. Sutskever. Learning Transferable Visual Models From Natural Language Supervision. In *38th International Conference on Machine Learning (ICML)*, 2021.

[15] O. Ronneberger, P. Fischer, and T. Brox. U-Net: Convolutional Networks for Biomedical Image Segmentation. In *International Conference on Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, 2015.

[16] A. van den Oord, O. Vinyals, and K. Kavukcuoglu. Neural Discrete Representation Learning. In *31st Conference on Neural Information Processing Systems (NIPS)*, 2017.

[17] E. Jang, A. Irpan, M. Khansari, D. Kappler, F. Ebert, C. Lynch, S. Levine, and C. Finn. BC-Z: Zero-Shot Task Generalization with Robotic Imitation Learning. In *5th Conference on Robot Learning (CoRL)*, 2021.

[18] V. Myers, A. He, K. Fang, H. Walke, P. Hansen-Estruch, C.-A. Cheng, M. Jalobeanu, A. Kolobov, A. Dragan, and S. Levine. Goal Representations for Instruction Following: A Semi-Supervised Language Interface to Control. In *7th Annual Conference on Robot Learning (CoRL)*, 2023.

[19] E. Perez, F. Strub, H. de Vries, V. Dumoulin, and A. Courville. FiLM: Visual Reasoning with a General Conditioning Layer. In *32nd AAAI Conference on Artificial Intelligence*, 2018.

[20] A. Brohan, N. Brown, J. Carbajal, Y. Chebotar, J. Dabis, C. Finn, K. Gopalakrishnan, K. Hausman, A. Herzog, J. Hsu, J. Ibarz, B. Ichter, A. Irpan, T. Jackson, S. Jesmonth, N. J. Joshi, R. Julian, D. Kalashnikov, Y. Kuang, I. Leal, K.-H. Lee, S. Levine, Y. Lu, U. Malla, D. Manjunath, I. Mordatch, O. Nachum, C. Parada, J. Peralta, E. Perez, K. Pertsch, J. Quiambao, K. Rao, M. Ryoo, G. Salazar, P. Sanketi, K. Sayed, J. Singh, S. Sontakke, A. Stone, C. Tan, H. Tran, V. Vanhoucke, S. Vega, Q. Vuong, F. Xia, T. Xiao, P. Xu, S. Xu, T. Yu, and B. Zitkovich. RT-1: Robotics Transformer for Real-World Control at Scale. In *Robotics: Science and Systems (RSS)*, 2023.

[21] M. Tan and Q. V. Le. EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks. In *36th International Conference on Machine Learning (ICML)*, 2019.

[22] M. Ryoo, AJ. Piergiovanni, A. Arnab, M. Dehghani, and A. Angelova. TokenLearner: Adaptive space-time tokenization for videos. In *35th Conference on Neural Information Processing Systems (NeurIPS)*, 2021.

[23] H. Bharadhwaj, J. Vakil, M. Sharma, A. Gupta, S. Tulsiani, and V. Kumar. RoboAgent: Generalization and Efficiency in Robot Manipulation via Semantic Augmentations and Action Chunking. (arXiv:2309.01918), 2023.

[24] T. Z. Zhao, V. Kumar, S. Levine, and C. Finn. Learning Fine-Grained Bimanual Manipulation with Low-Cost Hardware. In *Robotics: Science and Systems (RSS)*, 2023.

[25] H. Liu, L. Lee, K. Lee, and P. Abbeel. Instruction-Following Agents with Multimodal Transformer. (arXiv:2210.13431), 2023.

[26] X. Geng, H. Liu, L. Lee, D. Schuurmans, S. Levine, and P. Abbeel. Multimodal Masked Autoencoders Learn Transferable Representations. (arXiv:2205.14204), 2022.

[27] S. James and A. J. Davison. Q-Attention: Enabling Efficient Learning for Vision-Based Robotic Manipulation. *IEEE Robotics and Automation Letters (RA-L)*, 7(2), 2022.

[28] S. Karamcheti, S. Nair, A. S. Chen, T. Kollar, C. Finn, D. Sadigh, and P. Liang. Language-Driven Representation Learning for Robotics. In *Robotics: Science and Systems (RSS)*, 2023.

[29] R. Shah, R. M. Martın, and Y. Zhu. MUTEX: Learning Unified Policies from Multimodal Task Specifications. In *7th Conference on Robot Learning (CoRL)*, 2023.

[30] Y. Jiang, A. Gupta, Z. Zhang, G. Wang, Y. Dou, Y. Chen, L. Fei-Fei, A. Anandkumar, Y. Zhu, and L. Fan. VIMA: General Robot Manipulation with Multimodal Prompts. In *40th International Conference on Machine Learning (ICML)*, 2023.

[31] A. Brohan, N. Brown, J. Carbajal, Y. Chebotar, X. Chen, K. Choromanski, T. Ding, D. Driess, A. Dubey, C. Finn, P. Florence, C. Fu, M. G. Arenas, K. Gopalakrishnan, K. Han, K. Hausman, A. Herzog, J. Hsu, B. Ichter, A. Irpan, N. Joshi, R. Julian, D. Kalashnikov, Y. Kuang, I. Leal, L. Lee, T.-W. E. Lee, S. Levine, Y. Lu, H. Michalewski, I. Mordatch, K. Pertsch, K. Rao, K. Reymann, M. Ryoo, G. Salazar, P. Sanketi, P. Sermanet, J. Singh, A. Singh, R. Soricut, H. Tran, V. Vanhoucke, Q. Vuong, A. Wahid, S. Welker, P. Wohlhart, J. Wu, F. Xia, T. Xiao, P. Xu, S. Xu, T. Yu, and B. Zitkovich. RT-2: Vision-Language-Action Models Transfer Web Knowledge to Robotic Control. In *7th Conference on Robot Learning (CoRL)*, 2023.

[32] X. Chen, J. Djolonga, P. Padlewski, B. Mustafa, S. Changpinyo, J. Wu, C. R. Ruiz, S. Goodman, X. Wang, Y. Tay, S. Shakeri, M. Dehghani, D. Salz, M. Lucic, M. Tschannen, A. Nagrani, H. Hu, M. Joshi, B. Pang, C. Montgomery, P. Pietrzyk, M. Ritter, A. J. Piergiovanni, M. Minderer, F. Pavetic, A. Waters, G. Li, I. Alabdulmohsin, L. Beyer, J. Amelot, K. Lee, A. P. Steiner, Y. Li, D. Keysers, A. Arnab, Y. Xu, K. Rong, A. Kolesnikov, M. Seyedhosseini, A. Angelova, X. Zhai, N. Houlsby, and R. Soricut. PaLI-X: On Scaling up a Multilingual Vision and Language Model. (arXiv:2305.18565), 2023.

[33] P. Goyal, S. Niekum, and R. J. Mooney. PixL2R: Guiding Reinforcement Learning Using Natural Language by Mapping Pixels to Rewards. In *4th Conference on Robot Learning (CoRL)*, 2020.

[34] L. Shao, T. Migimatsu, Q. Zhang, K. Yang, and J. Bohg. Concept2Robot: Learning Manipulation Concepts from Instructions and Human Demonstrations. In *Robotics: Science and Systems (RSS)*, 2020.

[35] S. Nair, E. Mitchell, K. Chen, B. Ichter, S. Savarese, and C. Finn. Learning Language-Conditioned Robot Behavior from Offline Data and Crowd-Sourced Annotation. In *5th Conference on Robot Learning (CoRL)*, 2022.

[36] P. Mahmoudieh, D. Pathak, and T. Darrell. Zero-Shot Reward Specification via Grounded Natural Language. In *39th International Conference on Machine Learning (ICML)*, 2022.

[37] N. Di Palo, A. Byravan, L. Hasenclever, M. Wulfmeier, N. Heess, and M. Riedmiller. Towards A Unified Agent with Foundation Models. In *Workshop on Reincarnating Reinforcement Learning (RRL) at International Conference on Learning Representations (ICLR)*, 2023.

[38] A. Adeniji, A. Xie, C. Sferrazza, Y. Seo, S. James, and P. Abbeel. Language Reward Modulation for Pretraining Reinforcement Learning. (arXiv:2308.12270), 2023.

[39] S. Nair, A. Rajeswaran, V. Kumar, C. Finn, and A. Gupta. R3M: A Universal Visual Representation for Robot Manipulation. In *6th Conference on Robot Learning (CoRL)*, 2022.

[40] W. Huang, C. Wang, R. Zhang, Y. Li, J. Wu, and L. Fei-Fei. VoxPoser: Composable 3D Value Maps for Robotic Manipulation with Language Models. In *7th Conference on Robot Learning (CoRL)*, 2023.

[41] Y. J. Ma, V. Kumar, A. Zhang, O. Bastani, and D. Jayaraman. LIV: Language-Image Representations and Rewards for Robotic Control. In *40th International Conference on Machine Learning (ICML)*, 2023.

[42] P. Sharma, B. Sundaralingam, V. Blukis, C. Paxton, T. Hermans, A. Torralba, J. Andreas, and D. Fox. Correcting Robot Plans with Natural Language Feedback. In *Robotics: Science and Systems (RSS)*, 2022.

[43] A. Y. Ng, D. Harada, and S. Russell. Policy Invariance Under Reward Transformations: Theory and Application to Reward Shaping. In *16th International Conference on Machine Learning (ICML)*, 1999.

[44] M. Riedmiller, J. T. Springenberg, R. Hafner, and N. Heess. Collect & Infer - a fresh look at data-efficient Reinforcement Learning. In *5th Conference on Robot Learning (CoRL)*, 2022.

[45] E. Stengel-Eskin, A. Hundt, Z. He, A. Murali, N. Gopalan, M. Gombolay, and G. Hager. Guiding Multi-Step Rearrangement Tasks with Natural Language Instructions. In *5th Conference on Robot Learning (CoRL)*, 2022.

[46] A. Hundt, B. Killeen, N. Greene, H. Wu, H. Kwon, C. Paxton, and G. D. Hager. ”Good Robot!”: Efficient Reinforcement Learning for Multi-Step Visual Tasks with Sim to Real Transfer. *IEEE Robotics and Automation Letters (RA-L)*, 5(4), 2020.

[47] M. Shridhar, L. Manuelli, and D. Fox. CLIPort: What and Where Pathways for Robotic Manipulation. In *5th Conference on Robot Learning (CoRL)*, London, UK, 2021.

[48] A. Zeng, P. Florence, J. Tompson, S. Welker, J. Chien, M. Attarian, T. Armstrong, I. Krasin, D. Duong, A. Wahid, V. Sindhwani, and J. Lee. Transporter Networks: Rearranging the Visual World for Robotic Manipulation. In *4th Conference on Robot Learning (CoRL)*, 2022.

[49] M. Shridhar, L. Manuelli, and D. Fox. Perceiver-Actor: A Multi-Task Transformer for Robotic Manipulation. In *6th Conference on Robot Learning (CoRL)*, 2022.

[50] E. Johns. Coarse-to-Fine Imitation Learning: Robot Manipulation from a Single Demonstration. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2021.

[51] A. Jaegle, S. Borgeaud, J.-B. Alayrac, C. Doersch, C. Ionescu, D. Ding, S. Koppula, D. Zoran, A. Brock, E. Shelhamer, O. Hénaff, M. M. Botvinick, A. Zisserman, O. Vinyals, and J. Carreira. Perceiver IO: A General Architecture for Structured Inputs & Outputs. In *10th International Conference on Learning Representations (ICLR)*, 2022.

[52] A. Goyal, J. Xu, Y. Guo, V. Blukis, Y.-W. Chao, and D. Fox. RVT: Robotic View Transformer for 3D Object Manipulation. In *7th Conference on Robot Learning (CoRL)*, 2023.

[53] H. Ha, P. Florence, and S. Song. Scaling Up and Distilling Down: Language-Guided Robot Skill Acquisition. (arXiv:2307.14535), 2023.

[54] C. Chi, S. Feng, Y. Du, Z. Xu, E. Cousineau, B. Burchfiel, and S. Song. Diffusion Policy: Visuomotor Policy Learning via Action Diffusion. (arXiv:2303.04137), 2023.

[55] E. Zhang, Y. Lu, W. Wang, and A. Zhang. LAD: Language Augmented Diffusion for Reinforcement Learning. In *2nd Workshop on Language and Reinforcement Learning at NeurIPS*, 2022.

[56] M. Reuss and R. Lioutikov. Multimodal Diffusion Transformer for Learning from Play. In *2nd Workshop on Language and Robot Learning at Conference on Robot Learning (CoRL)*, 2023.

[57] J. Ho, A. Jain, and P. Abbeel. Denoising Diffusion Probabilistic Models. In *34th Conference on Neural Information Processing Systems (NeurIPS)*, 2020.

[58] Y. Song, J. Sohl-Dickstein, D. P. Kingma, A. Kumar, S. Ermon, and B. Poole. Score-based Generative Modeling Through Stochastic Differential Equations. In *International Conference on Learning Representations (ICLR)*, 2021.

[59] T. Karras, T. Aila, M. Aittala, and S. Laine. Elucidating the Design Space of Diffusion-Based Generative Models. In *36th Conference on Neural Information Processing Systems (NeurIPS)*, 2022.

[60] J. Song, C. Meng, and S. Ermon. Denoising Diffusion Implicit Models. In *International Conference on Learning Representations (ICLR)*, 2021.

[61] M. Janner, Y. Du, J. B. Tenenbaum, and S. Levine. Planning with Diffusion for Flexible Behavior Synthesis. In *39th International Conference on Machine Learning (ICML)*, 2022.

[62] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer. High-Resolution Image Synthesis with Latent Diffusion Models. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[63] I. Kapelyukh, V. Vosylius, and E. Johns. DALL-E-Bot: Introducing Web-Scale Diffusion Models to Robotics. *IEEE Robotics and Automation Letters (RA-L)*, 8(7), 2023.

[64] A. Ramesh, P. Dhariwal, A. Nichol, C. Chu, and M. Chen. Hierarchical Text-Conditional Image Generation with CLIP Latents. (arXiv:2204.06125), 2022.

[65] P. Sundaresan, S. Belkhale, D. Sadigh, and J. Bohg. KITE: Keypoint-Conditioned Policies for Semantic Manipulation. (arXiv:2306.16605), 2023.

[66] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng. NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis. In *European Conference on Computer Vision*, 2020.

[67] W. Shen, G. Yang, A. Yu, J. Wong, L. P. Kaelbling, and P. Isola. Distilled Feature Fields Enable Few-Shot Language-Guided Manipulation. In *7th Conference on Robot Learning (CoRL)*, 2023.

[68] A. Rashid, S. Sharma, C. M. Kim, J. Kerr, L. Y. Chen, A. Kanazawa, and K. Goldberg. Language Embedded Radiance Fields for Zero-Shot Task-Oriented Grasping. In *7th Conference on Robot Learning (CoRL)*, 2023.

[69] J. Kerr, C. M. Kim, K. Goldberg, A. Kanazawa, and M. Tancik. LERF: Language Embedded Radiance Fields. In *International Conference on Computer Vision (ICCV)*, 2023.

[70] Y. Zhou, S. Sonawani, M. Phielipp, S. Stepputtis, and H. B. Amor. Modularity through Attention: Efficient Training and Transfer of Language-Conditioned Policies for Robot Manipulation. In *6th Conference on Robot Learning (CoRL)*, 2022.

[71] F. Locatello, D. Weissenborn, T. Unterthiner, A. Mahendran, G. Heigold, J. Uszkoreit, A. Dosovitskiy, and T. Kipf. Object-Centric Learning with Slot Attention. In *34th Conference on Neural Information Processing Systems (NeurIPS)*, 2020.

[72] R. Wang, J. Mao, J. Hsu, H. Zhao, J. Wu, and Y. Gao. Programmatically Grounded, Compositionally Generalizable Robotic Manipulation. In *11th International Conference on Learning Representations (ICLR)*, 2023.

[73] M. Steedman. *Surface Structure and Interpretation*. Number 30 in Linguistic Inquiry Monographs. MIT Press, Cambridge, Mass, 1996. ISBN 978-0-262-69193-2.