A Hybrid Compositional Reasoning Approach for Interactive Robot Manipulation

Georgios Tziafas[1] and Hamidreza Kasaei[1]

***Abstract*— In this paper we present a neuro-symbolic (hybrid) compositional reasoning model for coupling language-guided visual reasoning with robot manipulation. A non-expert human user can prompt the robot agent using natural language, providing a referring expression, a question or a grasp action instruction. The model tackles all cases in a task-agnostic fashion through the utilization of a shared library of primitive skills. Each primitive handles an independent sub-task, such as reasoning about visual attributes, spatial relation comprehension, logic and enumeration, as well as arm control. A language parser maps the input query to an executable program composed of such primitives depending on the context. While some primitives are purely symbolic operations (e.g. counting), others are trainable neural functions (e.g. image/word grounding), therefore marrying the interpretability and systematic generalization benefits of discrete symbolic approaches with the scalability and representational power of deep networks. We generate a synthetic dataset of tabletop scenes to train our approach and perform several evaluation experiments for visual reasoning. Results show that the proposed method achieves very high accuracy while being transferable to real-world scenes with few-shot visual fine-tuning. Finally, we integrate our method with a robot framework and demonstrate how it can serve as an interpretable solution for an interactive object picking task, both in simulation and with a real robot. Supplementary material is available in this https URL.**

I. Introduction

As modern developments in robotics are beginning to move robots from purely industrial to human-centric environments, it becomes essential for them to be able to interact naturally with non-expert human users. This necessary feature poses additional challenges to traditional autonomy, as the agent should not only perceive and reason about its environment, but do so in a manner that is fully interpretable to its human cohabitants. Consider for instance the scenario presented in Fig. 1, where a user asks a question about a tabletop scene, referring to visual attributes and/or spatial relations between objects. Our intuition is that, for a human, the logic behind solving this task is compositional (a hierarchy of elementary steps) and disentangled from the actual scene content, meaning that the reasoning steps illustrated in Fig. 1 can be generalized to all similar questions regardless of the actual scene content.

Deep multi-modal learning methods for vision-language tasks such as referring expression comprehension (REC) and visual-question answering (VQA) tackle such challenges, however, due to their black-box nature, the desired compositional reasoning behavior can not be retrieved from the final answer, since it is often implicitly learned in the model representations.

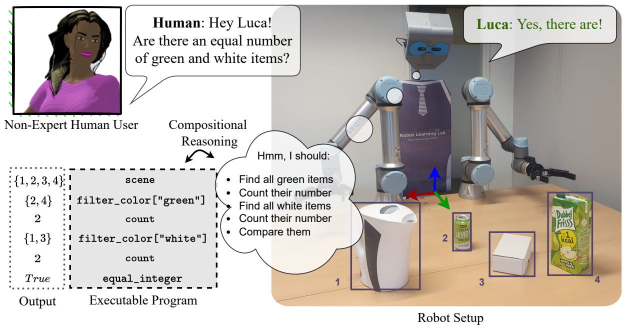


Fig. 1: Illustration of an HRI scenario where a robot interacts with a non-expert human user in free-form natural language. Understanding the input question / instruction often requires reasoning about properties or relations of appearing objects in a compositional manner. The robot parses the input question into the underlying reasoning program and executes it step-by-step in order to reach the final answer.

Therefore, such methods suffer from lack of interpretability as well as data-hungriness in order to be adapted to novel domains. To battle this, modern hybrid approaches combine deep representation learning with symbolic program synthesis over domain-specific primitives, an approach which is highly data-efficient and fully interpretable, due to disentangling discrete reasoning (symbolic) from perception (neural). However, prior arts ([13], [24]) usually study abstract domains with a poor variety of object and relation semantics, and fix their primitives to be aware of the domain vocabulary. In this work, we wish to propagate hybrid compositional reasoning approaches to the robotics domain and utilize it as an auxiliary process for interactive manipulation. Additionally, we reformulate components of the overall framework and design our primitives in an open-vocabulary fashion. In summary, the key contributions of this work are threefold:

- We develop a domain of synthetic tabletop scenes with a broad collection of object categories and rich spatial relation concepts and generate data for language-based visual reasoning. We further collect a real RGB-D dataset for evaluation, which we make publicly available.
- We propose a novel hybrid model for visual reasoning that uses domain-agnostic primitives and open-ended vision-language grounding, granting it transferable to novel scene content with minimal adaptation.
- We performed extensive experiments to show the merits of the suggested approach in terms of (*i*) high reasoning accuracy evaluated through a VQA task, (*ii*) data-efficient adaptation to real scenes and (*iii*) easiness of integration with a robot framework for an interactive object picking task, tested both in simulation and with a real robot.

[1]Department of Artificial Intelligence, University of Groningen, The Netherlands {g.t.tziafas,hamidreza.kasaei}@rug.nl

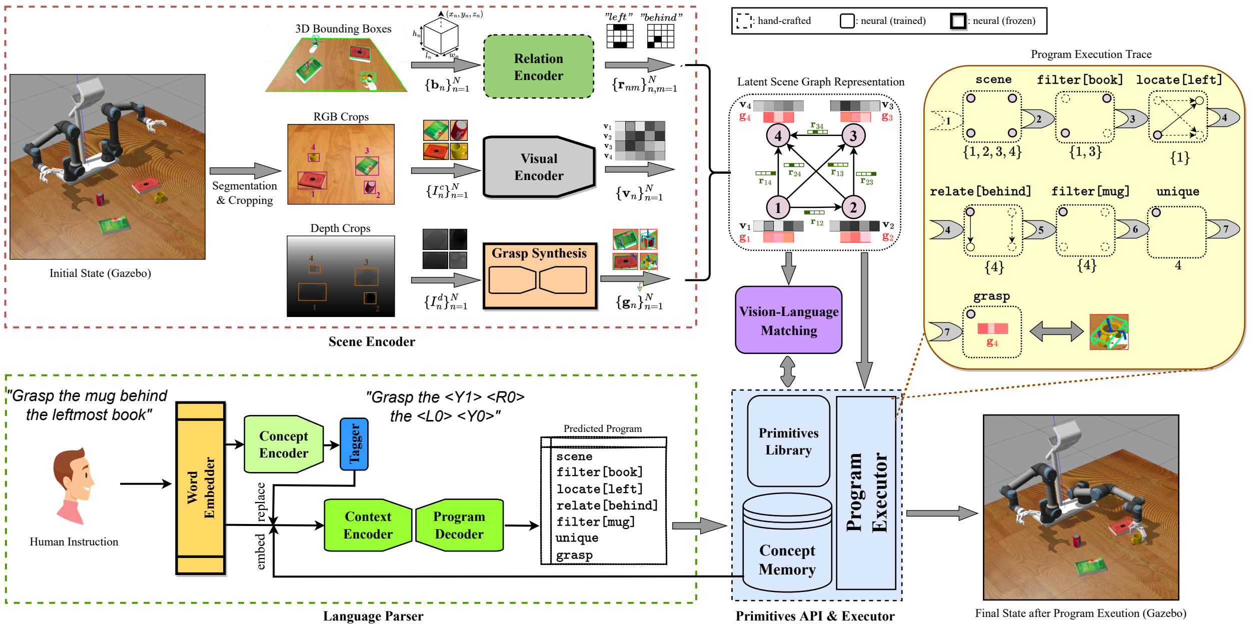


Fig. 2: A schematic of the proposed framework. First, objects are segmented and localized in 3D space *(top left)* and the scene is represented as a graph of extracted object-based features (visual, grasp pose) as nodes and their spatial relations as edges *(top middle)*. A human user provides an instruction and a language parser synthesizes an executable program *(bottom left)*, built out of a primitives library *(bottom middle)*. A program executor utilises a Vision-Language Matching (VLM) module to ground word concepts to different objects *(center)* and executes the predicted program step-by-step *(top right)*, in order to identify the queried object and instruct the robot to grasp it *(bottom right)*.

II. Related Work

Recent approaches for vision-language reasoning can be taxonomized [22] to holistic [7], [18], [21] and modular [1], [5], [10], [14]. Holistic approaches embed visual and text features into a common space and answer the question by fusing features, while modular decompose the task into sub-tasks treated by separate modules. The neuro-symbolic VQA model (NS-VQA) [24] introduces strong inductive biases for reasoning by integrating symbolic modules to pair the neural perception and language backbones. Symbolic modules are expressed as primitive operations within a Domain-Specific Language (DSL) and the language module parses the input query into an arbitrary composition of them as the underlying program. In this line of work however, the underlying scene is represented as a table of extracted attribute labels [24] or features [13], without any relation information. In our work, we integrate relation concepts with object-based features in a latent scene graph representation and make our primitives vocabulary-agnostic, allowing adaptation to novel concepts without the need for updating the DSL. Additionally, like NS-CL [13], we enable generalizable language parsing by replacing lexical items in the input query with their corresponding concepts, as defined in a concept memory. Unlike NS-CL, which applies hand-crafted methods for identifying concept values in the phrase, we learn the word to concept mapping through a word tagging module.

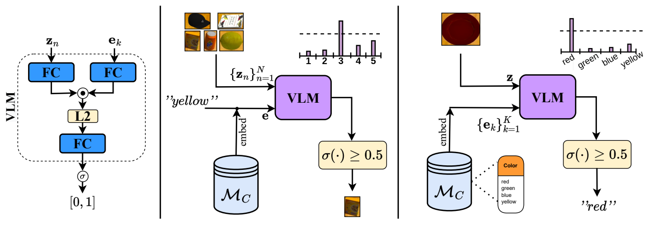
More recently, there have been attempts to adapt hybrid approaches for compositional reasoning in natural scene content [22] [8] [6]. In the robotics field, language-conditioning has been an emergent theme in RL-based manipulation [12]. In the overlapping space between compositional reasoning and manipulation, to the best of our knowledge, there are no published methods. SHOP-VRB [16] is a benchmark inspired from the CLEVR dataset [9], where the authors replicate the language - program pair generation for a kitchen domain of 20 objects with a rich variety of attribute concepts (weight, mobility etc.). Similarly, we adapt the CLEVR data generation engine to generate a dataset of synthetic tabletop scenes but extend it to include category and richer spatial relation concepts, absolute relations (e.g. *"The rightmost bowl"*) and higher-order (*hyper-*) relations (e.g.*"The bowl that is closer to the book than the mug"*).

III. Approach Description

Our architecture is comprised of four components: a) a scene graph encoder (hybrid), b) a language parser (neural), c) a dedicated language that implements reasoning / action primitives, paired with a program executor (symbolic) and d) a vision-language matching network (VLM - neural). Given a visual world state, the scene encoder constructs a scene graph representation which embeds object features as nodes and their spatial relations as edges. The language parser translates the input natural language query into the underlying program, and the program executor executes it as a sequence of message passing steps in the extracted scene graph. The VLM is used to match natural language phrases that express semantic concepts to appearing objects, serving as an open-ended alternative to classification. The overall framework with a running example are illustrated in Fig. 2.

A. Scene Graph Encoder

Given an input RGB-D pair of images, we first apply Mask-R-CNN [4] in RGB for instance segmentation and



(a) *VLM Module architecture* (b) `filter` *primitive example* (c) `query` *primitive example*

Fig. 3: A Vision-Language Matching (VLM) module (*left*) is used to ground word/phrase concepts to object instances and vice versa. Our program executor invokes VLM to perform filtering (*middle*) and querying (*right*) operations by computing matching scores for object-concept pairs. A Concept Memory $\mathcal{M}_C$ provides concept values and their embeddings to apply VLM in domain-specific vocabulary.

crop the N detected object instances from both frames $\{I_n^c \in \mathbb{R}^{h_n \times w_n \times 3}, I_n^d \in \mathbb{R}^{h_n \times w_n}\}_{n=1}^N$. Segmented objects are projected to 3D space using the camera intrinsics and approximated with a 3D bounding box $\mathbf{b}_n = \left(\frac{x_n}{L} \; \frac{y_n}{W} \; \frac{z_n}{H} \; \frac{l_n^x}{L} \; \frac{l_n^y}{W} \; \frac{l_n^z}{H} \right)^T$, where (W, L, H) denotes the workspace dimensions.

We then construct a scene graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}, \mathcal{X}_\mathcal{V}, \mathcal{X}_\mathcal{E}\}$ with nodes $\mathcal{V} = \{1, \ldots, N\}$, edges $\mathcal{E} = \mathcal{V} \times \mathcal{V}$, node features $\mathcal{X}_\mathcal{V} = \left\{\mathbf{x}_n^\mathcal{V} = (\mathbf{v}_n, \mathbf{g}_n),\ n \in \mathcal{V}\right\}$ and edge features $\mathcal{X}_\mathcal{E} = \left\{x_{nmr}^\mathcal{E} = \zeta_r(\mathbf{b}_n, \mathbf{b}_m),\ (n,m) \in \mathcal{E},\ r \in \mathcal{R}\right\}$. We extract the visual feature of each object by $\mathbf{v}_n = H(I_n^c)$, where $H : \mathbb{R}^{h_n \times w_n \times 3} \rightarrow \mathbb{R}^{D_v}$ is implemented by flattening the feature maps of up to the penultimate layer of a standard ImageNet-pretrained [3] CNN (we use *MobileNetv2* [20]). We parameterize grasping through 2D end-effector position, angle, openning length and grasp quality and utilize a pretrained vision-based grasp synthesis network (e.g. *GG-CNN* [15]), that receives the input depth crops I_n^d and generates a grasp proposal for each object $\mathbf{g}_n \in \mathbb{R}^5$.

For encoding relations, we define a vocabulary of spatial concepts $\mathcal{R} = \{$*"left", "right", "behind", "front", "closer", "further", "bigger", "smaller", "next to"*$\}$ and implement a heuristic function per concept $\zeta_r : \mathcal{E} \rightarrow \{0,1\}$ that utilises the 3D locations to decide whether the relation exists or not. For example, for deciding whether object n is left from m, we use the function $\zeta_{\text{"left"}}(n,m) = \left[x_n + \frac{l_n^x}{2} < x_m - \frac{l_m^x}{2} \right]$, where the $[\cdot]$ operator evaluates the input condition. Similarly, we calculate the relation value of all spatial concepts and aggregate them as the edge representation $\mathbf{x}_{nm}^\mathcal{E} \in \{0,1\}^{|\mathcal{R}|}$. All locations are expressed at the robot reference frame and so spatial relations are resolved from the robot's perspective.

B. Vision-Language Matching

Visual attributes refer to an object's properties (e.g. color, material) as well as its semantic category. In order to reason about such concepts, the agent needs to be able to ground the scene objects $n \in \mathcal{V}$ to specific concept values (e.g. *'bowl'* for category, *'red'* for color, *'plastic'* for material etc.) and vice-versa. We implement a module F^α that estimates a matching score between any given visual feature $\mathbf{v}_n$ of an object, and a concept embedding $\mathbf{e_c}$, encoded as the GloVe [17] embedding of the concept word c. The final matching score is given by:

$$F^\alpha(\mathbf{v_n}, \mathbf{e_c}) = \mathbf{\Theta_o^\alpha} \cdot \frac{\mathbf{\Theta_v^\alpha} \cdot \mathbf{v_n} \odot \mathbf{\Theta_e^\alpha} \cdot \mathbf{e}}{\left\|\mathbf{\Theta_v^\alpha} \cdot \mathbf{v_n} \odot \mathbf{\Theta_e^\alpha} \cdot \mathbf{e}\right\|_2} \tag{1}$$

where $\odot$ denotes element-wise multiplication and $\mathbf{\Theta_v^\alpha} \in \mathbb{R}^{D_v \times D_j}, \mathbf{\Theta_e^\alpha} \in \mathbb{R}^{D_q \times D_j}, \mathbf{\Theta_o^\alpha} \in \mathbb{R}^{D_j \times 1}$ are trainable matrices. Fig. 3 illustrates the architecture of the VLM network and how it is utilized to implement our visual reasoning primitives, namely filtering objects based on a concept value or querying for the concept value of a specific object. The set of all concept values and their embeddings(extracted for a given dataset) are maintained in the concept memory module $\mathcal{M}_\mathcal{C}$, which allows the VLM to query over all encountered concepts.

C. Language Parser

The language parser consists of two sub-modules, a concept tagger and a language-to-program encoder-decoder. We implement the concept tagger as a uni-layer Bi-GRU encoder of hidden size D_c (*ConceptEncoder*), followed by a projection $\mathbf{\Theta}_u \in \mathbb{R}^{D_c \times |\mathcal{M}_\mathcal{C}|}$ that maps hidden states to the most likely concept tag from the vocabulary included in concept memory $\mathcal{M}_\mathcal{C}$. We utilize pretrained GloVe [17]) word embeddings for representing words. After the tagging step we replace tagged words with the corresponding concept and add a unique index for that concept within the phrase (see example in bottom left of Fig. 2). The replaced sequence is fed to a seq2seq attention-enhanced encoder-decoder architecture [2]. A Bi-GRU encoder of hidden size D_π (*ContextEncoder*) encodes the input embeddings $\mathbf{e}_t$ to hidden states $\mathbf{h}_t^\pi = \texttt{Bi-GRU}(\mathbf{e_t}, \mathbf{h}_{t-1}^\pi)$, while the *ProgramDecoder* will autoregressively generate a sequence of primitive functions $\pi_t = \texttt{softmax}\left(\mathbf{\Theta}_\pi \cdot [\mathbf{h}_t^\pi; \mathbf{c}_t]\right)$ selected through greedy decoding from the primitives library Π, using a linear layer $\mathbf{\Theta}_\pi \in \mathbb{R}^{2D_\pi \times |\Pi|}$. Here, $\mathbf{c}_t = \sum_\tau \alpha_{t\tau} \mathbf{h}_\tau^c, \alpha_{t\tau} = \texttt{softmax}\left(\mathbf{h}_t^\pi \cdot \mathbf{\Theta}_{attn} \cdot \mathbf{h}_\tau^c\right)$ denotes the weighted average of the attention scores over the output states of the encoder. This formalism allows the decoder to recover attribute and relation concepts as inputs to primitives without having to define a new one for each concept-value pair, as in previous works [24]. In Sec. IV-A, we show that this enables zero-shot application of the parser in novel concept words, as long as their concept is tagged correctly.

D. Primitives and Program Execution

We define our library of reasoning primitives Π similar to the CLEVR domain [9], which we formally present in Table I. We further implement three primitives with additional semantics, namely: a) `scene`, which initializes an execution trace returning all objects $\mathcal{V}$, b) `unique`, which returns the object contained in a single-element object set, and c) `grasp`, which instructs the robot to grasp the object using the grasp proposal $\mathbf{g}_n$ of an input object n. Primitives are implemented as modules in a functional language, developed in Python. Our type system supports basic variable types, as well as two special types for representing an object and a object set through their unique indices in the scene graph nodes $\mathcal{V}$. The

TABLE I: The library of reasoning primitives included in our language, implemented as Python modules. For brevity we don't enumerate all combinations of primitive and concept arguments, but illustrate the latter as a separate column. *Visual* modules interface with the VLM and the scene's visual features to reason about visual attributes. *Spatial* primitives interface with the scene graph edge features to resolve spatial relations, superlatives (locations) and hyper-relations. *Symbolic* modules implement basic logic and arithmetics operations to incorporate integer and set semantics.

Reasoning	**Primitive**	**Concept Argument** (α)	**Type Signature**	**Semantics**	**Implementation**
	filter	Color, Material, Category	($\mathcal{V}_1$: ObjSet, c: `str`) $\rightarrow$ ObjSet	Returns subset of objects with given attribute concept value	$\{n \in \mathcal{V}_1 \mid \sigma\left(F^{\alpha}(\mathbf{v}_n,\ \mathbf{e_c})\right) \geq 0.5)\}$
visual	**query**	Color, Material, Category	n_1: Obj $\rightarrow$ `str`	Returns attribute concept value for given object	$\texttt{argmax}_c \left\{\sigma\left(F^{\alpha}(\mathbf{v}_1,\ \mathbf{e_c})\right),\ c \in \mathcal{M}_{\mathcal{C}}[\alpha]\right\}$
	same	Color, Material, Category	n_1: Obj $\rightarrow$ ObjSet	Returns subset of objects that have same attribute concept value with given object	$\texttt{filter}(\mathcal{V} - \{n_1\},\ \texttt{query}(n_1))$
	relate	Relation	(μ: Obj, r: `str`) $\rightarrow$ ObjSet	Returns subset of objects with given relation value to given object	$\left\{n \in \mathcal{V} \mid x^{\mathcal{E}}_{n\mu r} = 1\right\}$
spatial	**locate**	Relation	($\mathcal{V}_1$: ObjSet, r: `str`) $\rightarrow$ Obj	Returns object with most given relation values from given object set	$\texttt{argmax}_n \left\{\sum_{m \in \mathcal{V}_1} x^{\mathcal{E}}_{nmr},\ n \in \mathcal{V}_1\right\}$
	hyper_relate	Relation	(μ_1: Obj, μ_2: Obj, r: `str`) $\rightarrow$ ObjSet	Returns subset of objects with given relation value to given object pair	$\left\{n \in \mathcal{V} \mid x^{\mathcal{E}}_{n\mu_1 r} - x^{\mathcal{E}}_{n\mu_2 r} > 0\right\}$
	union, **intersection**	-	($\mathcal{V}_1$: ObjSet, $\mathcal{V}_2$: ObjSet) $\rightarrow$ ObjSet	Returns union/intersection of two given object sets	$\mathcal{V}_1 \cup \mathcal{V}_2,\ \ \mathcal{V}_1 \cap \mathcal{V}_2$
symbolic	**exist**, **count**	-	$\mathcal{V}_1$: ObjSet $\rightarrow$ `bool/int`	Returns size of given object set	$[\lvert\mathcal{V}_1\rvert > 0],\ \ \lvert\mathcal{V}_1\rvert$
	equal_integer, **greater**, **less**	Integer	(ν_1: `int`, ν_2: `int`) $\rightarrow$ `bool`	Compares two given integers	$[\nu_1 = \nu_2],\ [\nu_1 > \nu_2],\ [\nu_1 < \nu_2]$
	equal	Color, Material, Category	(c_1: `str`, c_2: `str`) $\rightarrow$ `bool`	Compares two given attribute concept values	$[c_1 = c_2]$

modules share the same type system and input/output interface and thus can be arbitrarily composed in any order and length. Whenever there is type mismatch between expected and retrieved inputs/outputs, the executor raises a suitable response, enforcing the interpretability principle by explaining to the user which reasoning step failed. This feature extends to cases of ambiguous queries (i.e. queried object appears more than once), as they will be similarly captured by the executor due to failure of the `unique` primitive type-checking (input set has more than one element).

E. Training Paradigm

The training process entails two optimization objectives: a) the correctness of the parsed program and b) word-object matching of the VLM module. Following insights from prior works [13], we train using a curriculum learning approach. In particular, we first train the VLM by isolating input/output pairs from execution traces of visual reasoning primitives in our dataset and express them as probability distributions over the node edges, optimized using binary cross entropy loss. For program synthesis, we first train the concept tagger on ground truth tags for a single epoch and then the entire language parser following [24]. First, we select a small diverse subset of our dataset and train using the ground truth programs and a softmax loss. Finally, we pair the language parser with frozen VLM and the program executor and train it on the remaining scenes with REINFORCE [23], using the correctness of the executed program as the reward signal.

IV. Experiments

In the subsections that follow we describe our experimental setup, the datasets and the evaluation results. We conduct our experiments in two datasets: a synthetic and a small-scale real-world RGB-D dataset.

*1) **Simulation**:* We collect from public resources a catalogue of 60 3D models from five types (fruits (6), electronics (6), kitchenware (17), books (5), stationery (10) and edible products (16)), organized into 26 category, 10 color and 8 material concepts. For evaluating in novel vocabulary, we also include special annotations for edible items according to their brand, variety or flavour (e.g. *"Coca-Cola"*, *"mango juice"*). We render synthetic scenes in the Gazebo environment [11] and generate around $8k$ training and $1.6k$ validation scenes, represented as symbolic scene graphs with all attribute-relation information. We develop on top of the CLEVR data generator [9] and extend the task templates to REC and grasping tasks, ending up with 13 task families, spawning a total of 289 templates. We instantiate 10 templates per scene and end up with around 80k training and $16k$ validation query-program-answer samples.

*2) **Real**:* In order to evaluate the adaptation performance of our model in natural scenes we record a dataset of **H**ousehold **O**bjects placed in **T**abletop **S**cenarios (HOTS). The object catalogue is a subset of the synthetic one but includes a few unseen categories and attributes, for a total of 48 object instances with 22 category, 9 color and 7 material concepts in 108 scene setups. We extract scene graphs and repeat the language-program-answer data generation step as above.

A. VQA Evaluation in Simulation

We report the overall accuracy of the executed programs in the dev set of our synthetic dataset, as well as in each question family separately. We compare out method with three holistic [10], [18], [21] and the original NS-VQA [24] baseline for VQA. The baseline models are trained with default hyper-parameters as mentioned in their paper and the DSL of NS-VQA is adapted for our synthetic domain. The final results are summarized in Table II. Out model is consistently above all holistic baselines across all question types, with the most significant margin in counting questions. Compared to vanilla NS-VQA, our approach achieves on-par performance, with a small drop due to the reformulation of our primitives library to be vocabulary-agnostic and the addition of the concept tagging bottleneck. When moving out-of-domain, the benefit of our formulation is shown in

TABLE II: VQA accuracy (%) per question type and overall for the dev split of our synthetic dataset.

Method	Count	Exist	Compare Number	Compare Attribute	Query	Overall
CNN-LSTM-SAN [10]	58.9	77.1	73.9	70.2	79.8	72.0
CNN-LSTM-RN [21]	86.3	93.7	87.05	91.6	92.8	90.3
CNN-GRU-FiLM [18]	88.3	93.4	89.35	92.9	93.2	91.4
NS-VQA [24]	98.6	98.1	97.8	96.4	97.1	97.6
Ours	97.6	97.0	96.9	96.1	96.3	96.8

TABLE III: VQA accuracy (%) for novel query instances, incl. novel attribute-category combinations (*Nov.Combs*) and unseen category words (*Nov.Words*). The NS-VQA baseline fails to parse unseen object descriptions due to its domain-aware formalism of primitives.

Method	Dev	Nov.Combs	Nov.Words
NS-VQA [24]	97.6	97.1	32.4
Ours (*GTtags + seq2seq*)	98.6	98.1	98.4
Ours (*ConceptTagger + seq2seq*)	96.8	97.0	80.9

Table III, where we evaluate the language parser in two test splits, including: a) novel combinations of attribute and category words, and b) unseen object category words. As expected, the vocabulary-specific baseline of NS-VQA fails to parse novel category concepts (as they are not part of the training primitives library), while our approach achieves significantly higher accuracy, with near-perfect results when evaluating only the seq2seq network with ground truth tags.

B. Adapting to Real Scenes

In this subsection we wish to assess the transferrability of our model to natural scenes by evaluating visual reasoning performance in the HOTS dataset. The modular nature of our approach allows us to bridge the sim-to-real gap solely in the vision domain, only adapting the VLM to real images from the HOTS dataset and transferring the language parser **without** any further training. We evaluate in two splits, namely: a) *HOTS-Perception*, where we only test the visual pipeline by treating attribute words as class labels as in recognition task, and b) *HOTS-Reasoning*, where we test the end-to-end system for REC and VQA tasks. For the first split, we use VLM for querying attribute concepts of input object images and report the percentage of correct top-1 predictions as accuracy. We initialize the VLM with the synthetic pretraining weights and fine-tune in different amounts of training examples per object instance (*1, 5, 20* and in *full* dataset). Results are summarized in Table IV. We observe that our method can be efficiently transferred to real scenes, as 20 labeled examples per object instance achieves very similar performance to fine-tuning in the entire dataset, both in the attribute recognition as well as in the end-to-end reasoning tasks.

TABLE IV: Top-1 accuracy (%) for classifying attributes - *category* (Cat), *color* (Col) and *material* (Mat) - as well as end-to-end REC and VQA tasks in annotated scenes of our HOTS dataset. *GT* denotes using ground truth perception. The @$\{1, 5, 20,\text{full}\}$ fields denote number of training images per object instance used to fine-tune the VLM module.

Method	HOTS-Perc.			HOTS-Reas.	
	Cat	Col	Mat	REC	VQA
GT + Parser	100.0	100.0	100.0	99.4	96.5
VLM@1 + Parser	43.2	44.4	60.8	49.7	47.7
VLM@5 + Parser	62.5	67.6	73.1	66.1	61.9
VLM@20 + Parser	90.5	89.9	94.4	89.8	86.6
VLM@*full* + Parser	93.4	91.8	95.7	90.9	88.1

TABLE V: Evaluating the system for coupling visual reasoning with robotic grasping in synthetic *(top)* and real *(bottom)* scenes.

Split	#Trials	#Failures	#Perc.Fail.	#Reas.Fail	#Grasp.Fail.
A	12	4 (33.0%)	1 (8.3%)	0 (0.0%)	3 (25.0%)
B	16	2 (12.5%)	1 (6.3%)	0 (0.0%)	1 (6.3%)
C	24	5 (20.8%)	4 (16.6%)	1 (4.2%)	0 (0.0%)
D	20	8 (40.0%)	5 (25.0%)	1 (5.0%)	2 (10.0%)
total	72	19 (26.4%)	11 (15.3%)	2 (2.8%)	6 (8.3%)
Real	12	3 (25.0%)	1 (0.0%)	2 (16.6%)	0 (0.0%)

C. Coupling Visual Reasoning with Robot Grasping

In this subsection we integrate our method with a robot framework and evaluate the end-to-end behavior for an interactive object picking task. An illustration of the setup and experiments is given in Fig. 4. In this round of experiments, we randomly place objects on a table and a human supervisor instructs the robot to grasp an object in real-time. The transparent nature of our method allows us to examine the program execution trace and diagnose the source of failures, including: a) *perception*, where there is either a localization error or the VLM has given an incorrect match, b) *reasoning*, where the parsed program is incorrect, or c) *grasping*, where the grasping fails (e.g. due to collision with obstacle).

For simulation, we report results in scenes separated in four splits, comprised of different levels of scene and query complexities, namely: **(A)**: scattered scenes (up to 5 objects) and simple queries (with programs up to depth 3); **(B)**: crowded scenes (up to 11 objects) with same query complexity, **(C)**: scattered scenes with queries of arbitrary complexity, and **(D)**: crowded scenes with queries of arbitrary complexity. For the real experiments, we conduct a total of 12 trials using objects from HOTS dataset the adapted visual pipeline of the previous section. Results are summarized in Table V. We observe that in both setups the averaged success rate is similar ($\sim 25\%$), with the reasoning module being robust to grasping instructions across all trials. Exception are a few queries in cases of complex question splits. Such failures are mostly due to the use of unknown spatial concepts by the human instructor (e.g. *between*). Perception errors occur more frequently in the crowded scene setup, due to partial views of objects leading to occlusion. The overall results showcase that the system can indeed serve as an accurate and interpretable interactive robotic grasper, as reasoning is demonstratively relatively robust to input scene and query complexity. We highlight that as our approach updates program execution by the output of perception in real-time, changes to the environment during execution will be captured within a given efficiency (around 4 Hz in our hardware setup), with the main bottleneck being the Mask-RCNN network.

V. Conclusion

In this work we bring together deep learning techniques for perception, grasp synthesis and NLP with symbolic reasoning in an end-to-end hybrid system, aimed for HRI applications. Results demonstrate that our method is highly accurate

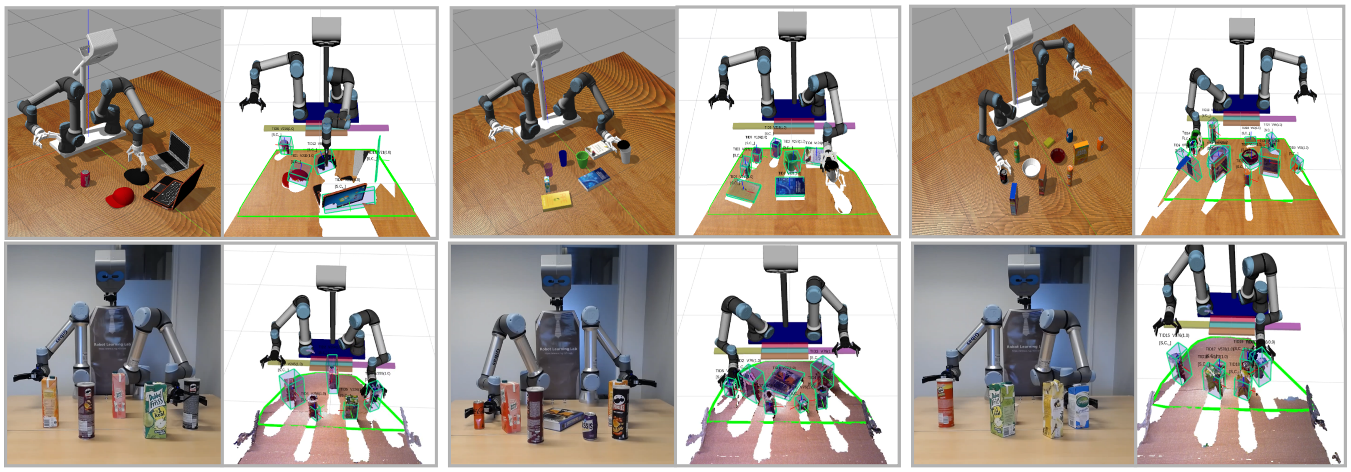


Fig. 4: A sequence of snapshots capturing the setup of our robot framework in Gazebo (*top*) and in a real-world environment (*bottom*). We generate a random scene and command the robot to grasp a specific item with a text instruction, using a disambiguation query. For our examples the queries are, from *top-left* to *right-bottom*: *"cap in front of largest laptop"*, *"plastic cup"*, *"rightmost soda can"*, *"leftmost Pringles"*, *"soda drink right from the pink juice box"*, *"rightmost item"*. In the snapshots we demonstrate the robot during the picking action (*left*) and the localization results in RViz (*right*).

and can be adapted to natural scene content with minimal adaptation. In particular, for adding new concepts, there is no fine-tuning required for the parser but instead an adaptation of the scene encoder (i.e., to include a new heuristic and edge representation for spatial) or the VLM (i.e., fine-tuning for visual). In order to alleviate the need for visual fine-tuning, in the future we plan to employ a foundation model (e.g CLIP [19]) for zero-shot visual-language grounding. For extending our approach to more primitives in order to deal with more complex reasoning and manipulation tasks, the new primitives must be formally defined and new synthetic data have to be generated to train the language parser. In the future, we plan to replace our supervised parser with a large language model for zero-shot code generation in unseen tasks.

References

[1] J. Andreas, M. Rohrbach, T. Darrell, and D. Klein. Deep compositional question answering with neural module networks. CoRR, abs/1511.02799, 2015.

[2] D. Bahdanau, K. Cho, and Y. Bengio. Neural machine translation by jointly learning to align and translate. ArXiv, 1409, 09 2014.

[3] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In 2009 IEEE conference on computer vision and pattern recognition, pages 248–255. Ieee, 2009.

[4] K. He, G. Gkioxari, P. Dollár, and R. Girshick. Mask R-CNN. In Proceedings of the IEEE international conference on computer vision, pages 2961–2969, 2017.

[5] R. Hu, J. Andreas, T. Darrell, and K. Saenko. Explainable neural computation via stack neural module networks. CoRR, abs/1807.08556, 2018.

[6] R. Hu, A. Rohrbach, T. Darrell, and K. Saenko. Language-conditioned graph networks for relational reasoning. CoRR, abs/1905.04405, 2019.

[7] D. A. Hudson and C. D. Manning. Compositional attention networks for machine reasoning. CoRR, abs/1803.03067, 2018.

[8] D. A. Hudson and C. D. Manning. Learning by abstraction: The neural state machine. CoRR, abs/1907.03950, 2019.

[9] J. Johnson, B. Hariharan, L. van der Maaten, L. Fei-Fei, C. L. Zitnick, and R. B. Girshick. CLEVR: A diagnostic dataset for compositional language and elementary visual reasoning. CoRR, abs/1612.06890, 2016.

[10] J. Johnson, B. Hariharan, L. van der Maaten, J. Hoffman, L. Fei-Fei, C. L. Zitnick, and R. B. Girshick. Inferring and executing programs for visual reasoning. CoRR, abs/1705.03633, 2017.

[11] N. P. Koenig and A. Howard. Design and use paradigms for gazebo, an open-source multi-robot simulator. 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (IEEE Cat. No.04CH37566), 3:2149–2154 vol.3, 2004.

[12] J. Luketina, N. Nardelli, G. Farquhar, J. N. Foerster, J. Andreas, E. Grefenstette, S. Whiteson, and T. Rocktäschel. A survey of reinforcement learning informed by natural language. In International Joint Conference on Artificial Intelligence, 2019.

[13] 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. CoRR, abs/1904.12584, 2019.

[14] D. Mascharka, P. Tran, R. Soklaski, and A. Majumdar. Transparency by design: Closing the gap between performance and interpretability in visual reasoning. CoRR, abs/1803.05268, 2018.

[15] D. Morrison, P. Corke, and J. Leitner. Closing the loop for robotic grasping: A real-time, generative grasp synthesis approach. CoRR, abs/1804.05172, 2018.

[16] M. Nazarczuk and K. Mikolajczyk. Shop-vrb: A visual reasoning benchmark for object perception. 2020 IEEE International Conference on Robotics and Automation (ICRA), pages 6898–6904, 2020.

[17] J. Pennington, R. Socher, and C. D. Manning. Glove: Global vectors for word representation. In In EMNLP, 2014.

[18] E. Perez, F. Strub, H. de Vries, V. Dumoulin, and A. C. Courville. Film: Visual reasoning with a general conditioning layer. CoRR, abs/1709.07871, 2017.

[19] 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. CoRR, abs/2103.00020, 2021.

[20] M. Sandler, A. G. Howard, M. Zhu, A. Zhmoginov, and L. Chen. Inverted residuals and linear bottlenecks: Mobile networks for classification, detection and segmentation. CoRR, abs/1801.04381, 2018.

[21] A. Santoro, D. Raposo, D. G. T. Barrett, M. Malinowski, R. Pascanu, P. W. Battaglia, and T. P. Lillicrap. A simple neural network module for relational reasoning. CoRR, abs/1706.01427, 2017.

[22] Z. Wang, K. Wang, M. Yu, J. Xiong, W.-m. Hwu, M. Hasegawa-Johnson, and H. Shi. Interpretable visual reasoning via induced symbolic space. In Proceedings of the IEEE/CVF International Conference on Computer Vision, pages 1878–1887, 2021.

[23] R. J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. Machine Learning, 8:229–256, 1992.

[24] K. Yi, J. Wu, C. Gan, A. Torralba, P. Kohli, and J. B. Tenenbaum. Neural-symbolic VQA: disentangling reasoning from vision and language understanding. CoRR, abs/1810.02338, 2018.