Embodied Reasoning for Discovering Object Properties via Manipulation

Jan Kristof Behrens[1,*], Michal Nazarczuk[2,*], Karla Stepanova[1,3], Matej Hoffmann[3], Yiannis Demiris[2], and Krystian Mikolajczyk[2]

***Abstract*— In this paper, we present an integrated system that includes reasoning from visual and natural language inputs, action and motion planning, executing tasks by a robotic arm, manipulating objects, and discovering their properties. A vision to action module recognises the scene with objects and their attributes and analyses enquiries formulated in natural language. It performs multi-modal reasoning and generates a sequence of simple actions that can be executed by a robot. The scene model and action sequence are sent to a planning and execution module that generates a motion plan with collision avoidance, simulates the actions, and executes them. We use synthetic data to train various components of the system and test on a real robot to show the generalization capabilities. We focus on a tabletop scenario with objects that can be grasped by our embodied agent i.e. a 7DoF manipulator with a two-finger gripper. We evaluate the agent on 60 representative queries repeated 3 times (e.g., 'Check what is on the other side of the soda can') concerning different objects and tasks in the scene. We perform experiments in a simulated and real environment and report the success rate for various components of the system. Our system achieves up to 80.6% success rate on challenging scenes and queries. We also analyse and discuss the challenges that such an intelligent embodied system faces.**

I. Introduction

AI systems for reasoning about visual scenes and answering nontrivial enquiries formulated in natural language have recently demonstrated impressive results [1], [2], [3]. Embodied scene reasoning capable of executing actions on real objects requires many components from the analysis of visual input to the hardware that performs actions in response to natural language queries. Much progress has also been made in the fields that focus on individual components such as visual recognition [4], 3D scene modelling [5], [6], NLP systems [7], [8], grasp prediction [9], action and motion planning [1], [10], modelling physical properties of objects [11] and their manipulation [12]. Apart from simulated environments [13], [14], there have been few studies integrating these components into a complete system with robotic hardware that would allow to identify the remaining bottlenecks.

[1]J. K. Behrens and K. Stepanova are with the Czech Institute of Informatics, Robotics, and Cybernetics, CTU in Prague, CR [jan.kristof.behrens, karla.stepanova]@cvut.cz

[2]M. Nazarczuk, Y. Demiris and K. Mikolajczyk are with the Department of Electrical and Electronic Engineering, Imperial College London, London, UK [michal.nazarczuk17, y.demiris, k.mikolajczyk]@imperial.ac.uk

[3]M. Hoffmann and K. Stepanova are with the Dep. of Cybernetics, Faculty of Electrical Engineering, CTU in Prague, matej.hoffmann@fel.cvut.cz

[*]Authors contributed equally

(a) Real scene setup. (b) Scene in Blender. (c) Scene in MuJoCo.

Fig. 1: Example scene based on real robot setup. Images include the view of the real scene: (a) rendered in Blender (b), and the corresponding setup in MuJoCo simulator (c).

In this paper, we propose a system that integrates a reasoning apparatus with an embodied agent in simulated and real environments. It is capable of understanding human formulated tasks, disambiguating scene objects, planning actions and motions, manipulating real objects, as well as measuring their physical properties (e.g., weight, stiffness).

With the proposed system, we address a number of important design questions for implementing an embodied agent. In particular, the reasoning system (see *V2A model* in Fig. 3) integrates components that are trained from existing datasets [15] or generated by simulators [16], which allow the system to generalise to new environments (including all test scenes used in our experiments). Our approach segments a long-term goal into atomic action sequences, which can be executed by a robot to acquire information about the environment via object manipulation. In this way, the embodied agent can improve its model of the environment and subsequently improve planning and execution of more complex tasks. The loop is closed by an action feedback and knowledge fusion pipeline that accumulates the knowledge for further, more efficient manipulations. In this work, we focus on one direction of the interaction: from visual input, over reasoning, to manipulation actions refining current knowledge. A closed loop triggering reasoning iterations remains a future work.

During object manipulation, humans acquire knowledge about the object properties like mass and stiffness on the fly. We propose suitable representations for objects, scenes, actions, and perceptions to foster successful execution and scalable knowledge acquisition. We select the necessary components and interfaces that facilitate the interplay between high-level action reasoning and low-level control schemes such as open-loop execution with sanity checks. We also address the robot/embodiment requirements which are hardware specific (e.g., workspace, kinematics, gripper collisions, etc.).

Finally, we evaluate the success of reasoning, motion planning, and execution of the system components on a set

of complex tasks (e.g., *"stack the yellow bowl behind the glass on top of the small plastic object that is on the right of the big container."*). In particular. We measure the success of completing the tasks and discuss the reasons for failures that originate from different components. We report how often and why action executions fail when performed by a real or simulated agent. We discuss the assumptions required for successful execution and how the success rate is affected by uncertainty in the scene representation.

In summary, we make the following contributions:

- We build an embodied reasoning agent, trained from synthetic data, but capable of generalizing to different tasks, objects and scenes.
- We propose a system for exploring object properties that require manipulation.
- We propose an evaluation framework, report the success rates for critical components, and identify weaknesses of the embodied agent.

We provide the source code for robotic actions, scene and object descriptions, generated action sequences, and other supporting material (`http://imitrob.ciirc.cvut.cz/EQA.html`) that can form a benchmark for similar systems.

II. Related work

There has been an increased interest at the intersection of robotics, computer vision, and natural language processing. There are many research activities in various areas including the direction of information (unidirectional, for example, from human to robot [17], [18], or bidirectional [19]), the purpose of linguistic interaction (e.g., to provide instructions [17], [20], [21], [22], [18], explanations [23], or obtain further information [1], [2], [3]), the degree of embodiment (from virtual agents [1], [2], [3] to real robots [17]), as well as the complexity of the language used (from simple utterances, to complex constructs), which dictates the level of reasoning that the listener has to perform to understand the linguistic construct. To facilitate research in these areas, a range of related datasets have been created, including SHOP-VRB [16], ALFRED [14], EQA [2], and R2R [24].

Fewer studies focused on building a complete system that can understand and execute human queries which require manipulation of real objects. Simulated environments [13], [14] are convenient and efficient for performing experiments, but cannot capture all the possible challenges that working with the real hardware poses. We extensively use synthetic data to train the components of the reasoning system [1] and evaluate it in new virtual and real scenes. We advance the state-of-the-art in this area by integrating various components together. This allows to evaluate their performance and readiness for deployment in an embodied reasoning system.

III. Embodied reasoning system

In this section we present our embodied reasoning system: the scenes and objects used to perform the experiments, the reasoning module, action primitives, and their parameters. More complex tasks are discussed at the end. Figure 2 shows

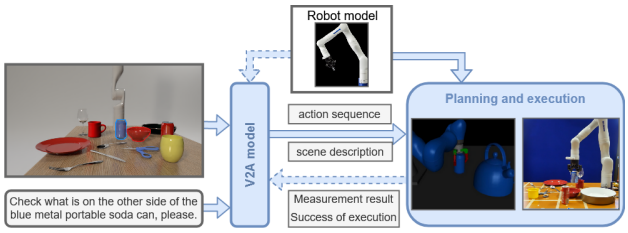

Fig. 2: The main components and information flow in the embodied reasoning system. Blue boxes are system components, dark grey boxes mark input data, blue arrows show data flow (with light grey boxes showing the data type). Dashed arrows mark information links which are established, but the receiving component operates independent of the data. This is due to the fact that the query is finished when the measurement data arrives.

the main system components and the information flow between them. Visual scenes and queries formulated in natural language are the inputs to the Vision to Action module [1]. It generates the scene and object description together with the sequence of actions that should be executed to answer the query. These are passed to the planning and execution module that uses MoveIt! [25] to plan individual actions with collision avoidance. The actions are then executed in a simulation environment or on the real robot. The execution module reports the measurements and action feedback to the reasoning module.

A. Synthetic Scenes and Questions

We adapt the methods from SHOP-VRB [16] and V2A [1] to generate randomised scenes and questions for the experiment, making the following adjustments to match real and simulated scenes. The base of the scene is changed to a table and the model of Kinova Gen3 robotic arm is added. Additionally, camera parameters (position, orientation and field of view) in Blender [26] are adjusted to resemble the real configuration (see Fig. 1). Scenes with random objects are then generated. We use the set of objects that are present in both SHOP-VRB and YCB dataset and adjust the synthetic objects (materials, colours) to resemble the YCB objects. The following items were used to generate the scenes: bowl, fork, glass, knife, mug, pan, plate, scissors, soda can, spoon, and wine glass. For the experiment with the real setup, we use the scene containing all the objects with an extra soda can in a different colour shown in Fig. 1(b). The questions were generated via V2A [1] generator filtering only those with a valid answer.

B. Vision to Action Reasoning

Our approach to reasoning builds on the V2A model [1], presented in Fig. 3 and consists of three main components: scene segmentation, attributes extraction, and multi-modal reasoning on both instructions and scene representations. Scene segmentation is performed using Mask R-CNN [4], which predicts the segmentation mask and category of each object in the scene. Thereafter, masked images of objects are passed to ResNet-34 [27] responsible for extracting the

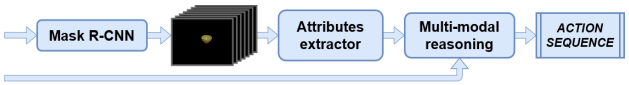

Fig. 3: V2A model used for the visual and NLP reasoning.

TABLE I: Subset of primitive actions chosen from V2A.

Command	Functioning
avoid:IDX	Avoid object IDX during next move.
move:IDX	Move gripper over the object IDX.
move:DIR	Move effector in cardinal direction DIR.
approach grasp:IDX	Approach grasping position to object IDX.
grasp:IDX	Close the gripper (called after approach grasp).
release	Release object from the gripper.
measure weight	Measure weight of the object in the gripper.
measure stiffness	Measure stiffness of the object in the gripper.
shake	Shake the object in the gripper.
rotate	Rotate the object in the gripper (around z axis).
flip	Flip the object in the gripper upside down.

attributes of the given items, providing a disentangled representation for the properties of the objects. Action sequence prediction is based on Seq2seq machine translation [28]. The instruction and the sequence of object attributes are encoded to a latent space with a LSTM [29] network. As in V2A, a multi-modal attention module is then applied to outputs of both encoders. Finally, given the attention map, the sequence of primitive actions is predicted using an LSTM decoder initialised with the hidden state of both encoders (concatenated). The list of primitive actions is presented in Table I. V2A model is fully trained on the dataset from [1] and is not further tuned for the scenes used in this work.

C. Actions and their Parameters

The primitive actions enable to operate the robot relative to objects in the scene, the current pose, and approach an object for grasping or other measurement actions. Most primitive actions (see Table I) are parameterized for the actual scene.

Below we provide details for selected actions from Table I.

1) Move: moving the end-effector above the object specified by the parameter IDX. Motions are checked against collisions with objects specified by the *avoid* primitive. *Factors affecting outcome:* Object can be placed out of reach or other scene objects obstruct the motion.

2) Grasp: Grasping is achieved by the combination of the primitives *approach grasp* and *grasp*. The first moves the end-effector in a Cartesian motion from the current pose into one of the pre-defined grasp poses (all options are tried until a feasible motion plan is found). The *grasp* action closes the gripper and monitors the gripper forces and motion to infer if the action was successful. *Factors affecting outcome:* Errors in the estimated object pose can lead to a collision between the gripper and the object or to unstable grasps. Other objects can block the approach, such that no motion plan can be found. When using a trained model for grasp selection (e.g., [30]), failures can be induced by bad choices.

3) Measure weight: With the object grasped, the robot moves to a specific joint configuration in which the object's weight projects into specific robot axes due to gravity. The torques are measured and compared with the readings when no object is lifted and the difference together with the lever arm length can be used to compute the object mass. *Factors affecting outcome:* The skill depends on the knowledge of the lever arm and the torque without payload τ_0. The lever arm depends on the grasp pose and thus can affect the estimated mass. τ_0 depends on the mass distribution in the measuring pose, but might be affected by friction and slack in the joints.

4) Measure stiffness: Close the gripper until in contact with the object. Then press with higher force and record the achieved deformation. *Factors affecting outcome:* Each gripper and robot have different sensory feedback. Some form of force / tactile feedback is necessary and the procedure needs to be calibrated for the current hardware. Relative ordering by stiffness is feasible. Measuring absolute physical quantities (stiffness / elasticity) needs a calibrated gripper.

D. Execution of Action Sequences

Fig. 4 shows the *execution pipeline*, its inputs (action sequence, scene and object descriptions), and its outputs (robotic execution, measurement results, and success reports). Each action of the sequence corresponds to a small robot program that is parametrized by the pipeline input (i.e., parameters and scene/object descriptions). We generate executable Python code from the action templates, which (*i*) implements the robot action (utilizing the manipulator, the gripper, and cameras), and (*ii*) collects and evaluates the sensor data to report the actions' results and detect possible failures. The real and simulated hardware can be used interchangeably thanks to compatible interfaces.

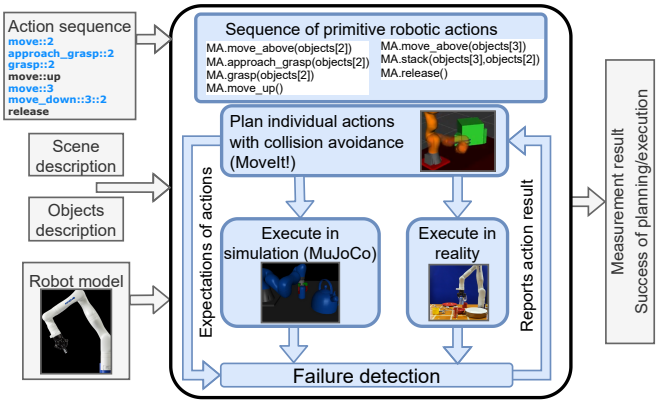

Fig. 4: Executing action sequence from V2A with simulated or real robotic arm.

IV. Experimental Setup

We evaluate the individual modules as well as the complete system in simulation and real experiments. First, we elaborate the simulated and the real robot setup Then, details about the conducted experiments are given Lastly, the evaluation metrics are defined.

A. Real Robot Setup

The real robot setup is illustrated in Fig. 1(a). It consists of a Kinova Gen3 robot manipulator (7 DoF) with a Robotiq 2F-85 gripper on a table. The robot has joint torque sensors in every axis that are exploited to estimate the mass of objects

TABLE II: Benchmarking tasks used in experiments.

#	Task
1	Pick the OBJ up.
2	Pick the OBJ and move it to the DIR.
3	Pick the OBJ1 and move it over OBJ2.
4	Measure weight of the OBJ.
5	Measure stiffness of the OBJ.
6	Empty OBJ (pick it up and flip upside down).
7	Check what is on the other side of the OBJ (pick up and rotate).
8	Shake the OBJ to check if there are any moving parts.
9	Put OBJ1 in the OBJ2.
10	Stack OBJ1 on top of OBJ2.

the robot is lifting. The gripper is equipped with position feedback (gripper aperture) and motor current feedback, which enables to estimate the object stiffness. Two RGB-D cameras (Intel Realsense D435) are mounted on the opposite side of the table viewing the scene.

B. Simulation Setup

The simulation environment consists of two parts: we simulate the robot, the scene objects, and their physical interaction using MuJoCo [31]. The 3D rendering software Blender [26] is used to create photo-realistic images from the scene as seen from the viewports of two Intel Realsense D435 cameras (see Fig. 1(a)) in the setup.

The MuJoCo simulation model is specified as Universal Robot Description Format (URDF), which allows to describe kinematic structure composed of rigid bodies (links) and their kinematic constraints (joints). The robot model (including the gripper and the table) is equivalent with the configuration of the real robot. The objects are *connected* to the world frame via floating joints, which do not restrict the object's motions.

The robot is controlled by a ROS-Control [32] effort controller with a custom hardware interface. We tuned the PID controller parameters to allow position control of the robot and joint trajectory control. The latter is exposed as joint trajectory action, which allows us to control the simulated robot using the MoveIt! motion planning framework [25].

The state of the objects in the MuJoCo environment is determined by the physical simulation taking into account the contact states, the resulting forces, the object geometry, and the mass properties of the objects. For a stable simulation, we had to simplify and clean the meshes (e.g., approximation by convex decomposition). For details, see our website.

C. Visual Reasoning Experiments

We generated artificial test data (i.e., 1000 new questions with balanced distribution of tasks according to Tab. II) for the scene shown in Fig. 2 (example of the generated sequence is in Fig. 4) and tested the performance of the V2A method [1] (see Fig. 3). We then replicated the scene with the real robot and recorded images with the real cameras. Both experiments were conducted with the ground truth scene data as the input for the reasoning system, and using the full pipeline including the segmentation and attribute extraction modules.

D. Real-Robot Experiments

For evaluation of the system with the real robot, for each scene we selected 50 action assessed as correct by the V2A system. We tested our system on two different scenes to evaluate its robustness with respect to the experimental setup (see our website `http://imitrob.ciirc.cvut.cz/EQA.html` for the Scene 1 input data). We tested each action sequence for feasibility by a teleoperated execution. Then we let our system plan and execute the action sequences 3 times automatically. This resulted in 50x3 executions for Scene 1, and 10x3 executions for Scene 2. An example of a successful execution can be seen in Fig. 5.

E. Evaluation Metrics

We report global success rates and analyse specific failure modes for the experiments with the real setup. We report the overall success rate for all executed 180 action sequences as well as the success rates of the system components (V2A, planning, execution). We analyse individual action sequence executions with respect to various reasons for failure:

1) Collisions: We consider 4 types of collisions: 1) Unwanted collision with the target object during grasp approach (due to inaccurate pose estimation). 2) Collisions with neighbouring objects during pick up actions occur due to orientation changes of the grasped object when the gripper closes. Then a collision may not be detected, but it can be *acceptable* as nothing can be clamped during a pure upward motion. We utilise bounding boxes as collision bodies. In the simulation, collisions checks with the object are disabled during grasping. 3) Grasping positions may be obstructed by other objects. Our teleoperation serves as a test that all our sequences are possible to execute without such collisions. 4) Other types of collisions occur between robot and objects (e.g., collision of the object attached to the gripper with other objects during general movement).

2) Grasping: Objects are not properly grasped and slide out of the gripper during execution.

3) Action outcome not fulfilling the purpose: The action execution was completed without failures, but the intended result was not achieved. For example, the information on the hidden side of an object is not visible after rotating the object; or, stacked objects do not stay on top of each other. Sometimes, the action achieves the correct result, but the action execution violated common sense expectations—e.g., releasing an object too high above the container and thus dropping it instead of placing it carefully. Possibly, the reasoning could give some information about the purpose of the action. Establishing a feedback loop can help to increase the chance of success.

4) Measuring error: For measuring actions such as weighing and stiffness measurement, the final value may not correspond to the real object property. This may be result from wrong execution of the action or other effects (e.g., object grasped on the side instead of the centre).

(a) move::7 (b) approach::7 (c) grasp::7 (d) move::up (e) move::0 (f) down::0::7 (g) release

Fig. 5: Example of executing action sequence for *"Could you put the small blue portable object inside the medium-sized red ceramic irregularly shaped thing, please?"* For more examples, see the accompanying video and our website.

TABLE III: Results of visual reasoning and action planning. GT and EVAL refer to the use of either ground truth or inferred attributes, respectively. We consider instructions with valid action sequence and compare them to the simulated scenes as well as to the real photos of the real scene.

Data	Attributes	Success rate [%]
V2A [1] - valid	GT	24.9
V2A [1] - valid	EVAL	23.8
Real setup sim	GT	26.7
Real setup sim	EVAL	25.1
Real photos	EVAL	17.4

V. Experiments and Results

We present the results achieved for individual modules of our system as well as overall success rate of execution. We also provide a discussion on different failure types.

A. Visual Reasoning and Action Planning

The results for reasoning and action planning are presented in Table III. As we generated only instructions that have a valid action sequence as possible output, we consider only the valid part of the original V2A for comparison. We observe that with the new simulated scene, the results are comparable to V2A results from [1]. A slight improvement may be due to the reduced set of viable tasks, which would otherwise decrease the score. We notice a 30% decrease in performance when using real photo as the input. It is mainly the result of degraded Mask R-CNN performance due to lack of training on the real scenes. From the predictions we have observed that the cutlery was often missed by the detector. Similarly, the background clutter posed a challenge to the segmentation network. The second contribution to the error is the performance of the attribute extraction network. However, as questions are posed using only selected attributes, misclassifications of attributes did not have a significant impact on the full pipeline performance.

B. Evaluation of Measuring Actions

We performed qualitative and quantitative evaluation to show the dependency of individual actions on the experimental setup or the object pose.

1) Mass: We measured ten times the mass of a full and an empty soda can. The full and empty soda can were estimated to have a mass of 352 ± 4 g and 34 ± 12 g, respectively (355 g vs. 14 actual mass – reference). The variance in the weight measurements is correlated with the position of the can in the gripper (see Fig. 6), as the lever l is a divisor in the weight calculation $m = (\tau_0 - \tau)/(l * g)$. Another, less obvious, influence has the joint temperature. We observed

Fig. 6: The result and the perception of an embodied action depends on the hardware and the spatial configuration. The result of the weighing action for the shown object poses differ by 15 % relative to the nominal weight.

TABLE IV: Measure stiffness for different objects and grasps. The grasp types: YCB: full (1) vs. half (2) surface in contact with the gripper; Plastic cup: outside grasp (1) vs. around side (2); Can: Top (1) and middle (2) grasp.

Grasp	Can (empty)	Can (full)	YCB block	Plastic cup
1	0.7 ± 0.2	0.52 ± 0.32	3.64 ± 0.34	1.9 ± 1.1
2	1.8 ± 3.8	0.65 ± 0.52	4.78 ± 0.66	0.7 ± 0.7

τ_0^{25} to be 9.3017Nm and at constant operation temperature of 39.8 deg C $\tau_0^{39.8} = 9.558$Nm. This difference itself would account for a difference of 83 g in the estimated weight.

2) Stiffness: We executed sequences of 10 consecutive measurements with 6% and 40% maximum force on an empty can, a full can, the YCB rubber block, and a plastic cup. On each object, we selected two different grasps to apply the pressure (see Fig. 7). The results are presented in Table IV. For objects with sufficient elasticity (e.g., the YCB block and the plastic cup) the method returns reliable values with a small standard deviation. In the case of the empty can, we observed a significant deformation in the first trial (13%), but nearly no deformation in the subsequent ones due to plastic deformation. The perceived deformation is strongly dependent on the actual grasp, the object geometry, and distribution of material properties. Together, these factors determine the deformation modes and amount for given forces applied by the gripper (see Fig. 7 for different situations/grasps on the YCB block and a plastic cup). However, the perceived deformations are still suitable to classify the objects as *soft* or *hard*.

C. Motion Planning and Robot Execution

All the selected action sequences were tested for feasibility and executability (e.g., executable without collisions) by a

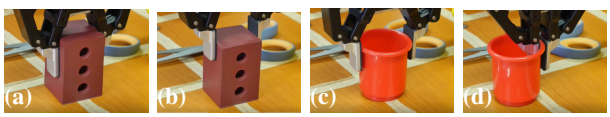

Fig. 7: Stiffness measurement depends on several factors such as grasp position (a,b) or grasp type (c,d).

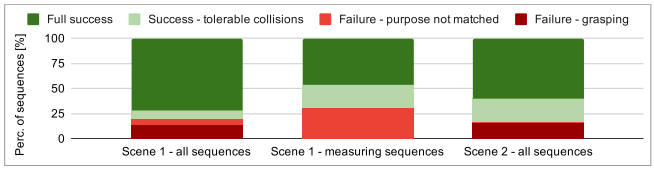

Fig. 8: Distribution of experiment results for Scene 1 (150 sequence, including 33 measuring sequences) and Scene 2 (30 sequences). We differentiate between failure due to grasping and failure where the outcome is not matching the expectations. Successful executions are divided to full success and success with tolerable collisions.

teleoperation test—a human operator commanded the robot to perform the given action.

For all primitive actions apart from *approach grasp* and *grasp*, we were able to always find collision-free motion plans. Collision-free grasp approaches were found for 9 out of 11 objects. The results for all real robot executions are shown in Fig. 8. We observed similar failure distribution over all the executions in both scenes. For Scene 1 (with 50 action sequences repeated 3 times), 80.6% of the executions were successful. Out of these, 71.3% were fully successful and in 9.3%, tolerable collisions occurred (e.g., touching neighbouring objects while moving up). In 6% of the cases, the action sequence was completed without errors, but the outcome did not match the action purpose. Failures due to wrong grasping or crashing objects happened in 13.3% of the executions. For Scene 2 with 10 action sequences (again repeated 3 times), we achieved similar results—full success in 60%, success with tolerable collisions 23.3%, and failure due to wrong grasping in 16.7%.

The outcome of the action did not match its purpose in the following cases: 1) action *rotate*: for 1 of 3 objects, the object was rotated outside of the camera field of view, making the label unreadable. 2) action *stack*: in 5 out of 10 action sequences executions, the object was not standing on top of the object to be placed. Sometimes, we observed action sequences, where the action's final outcome was matching (i.e. the object was on/in the other object), but the execution violated the common sense of how the action should be performed (e.g., action *stack* let the object fall in 33%, action *flip* spilled in 20% of cases the content over the robot).

Grasping failures (e.g., the object falls out of the gripper) were observed mainly during the manipulation of the knife (50% of cases) and the scissors (66% of cases). Grasping of the other objects did not work all the time, but these errors were already counted as collisions during the grasp approach. Tolerable collisions were observed for e.g., grasping a mug (touching the wine glass) and grasping the blue can (touching bowl). Furthermore, we observed collisions when lifting the scissors or cutlery up, because the orientation in the gripper of these objects slightly changed during grasping.

Sliding motions of the knife in the gripper and changes in the scissors' opening angle led to too low stiffness classifications for these objects. Weighing showed a measuring error of more than 50% with high standard deviation for the empty can, which was caused by the weak signal to noise ratio for weighing a 13g light object with an approximately 10kg heavy machine.

VI. Conclusions

We proposed an embodied reasoning system capable of planning and executing action sequences to answer complex queries formulated in natural language. Our system can be extended to disambiguate abstract instructions such as *Give me the full bottle, please!* or *Give me the softest object* which may require sequential exploratory acting to enable further reasoning as inferring a definite set of actions may not be possible from the initial knowledge and state of the scene. For example, to disambiguate a question related to fullness, the system will need to sequentially pick up and measure the weight of all bottles until a *full bottle* is found. To propose appropriate actions that can answer such query, linguistic constructs (e.g., full, soft, heavy, etc.) need to be associated to the type of actions that can provide the necessary sensory data. If the instructions involve ambiguous targets (i.e. several objects can be referenced), the system should be able to produce an action sequence that can acquire a differentiating piece of information (e.g., the weight of all bottles). A set of required sensor readings can be obtained by interacting with the environment. The planning system can then update its knowledge base and either answer the original query or issue more intermediate goals. We demonstrated that such systems can be integrated from various components such as scene parsing, natural language processing, reasoning, action planning with collision avoidance, simulation as well as real hardware execution. In addition, we provide an analysis and make a number of observations concerning the integration of the simulated and real environments that will facilitate the development of an improved intelligent robotic platform.

The success rate of completing the action sequences and providing answers to the queries is up to 80.6% which we consider very high given the complexity of the tasks. There are still many subtle challenges that compromise the quality of result and more engineering is needed on the level of action primitives and their parametrization to achieve robustness. Presented success rates emphasise the difficulties in predicting a whole sequence of actions. Incorrect predictions arise from the challenging task of identifying which item is the subject of the consecutive actions. Nonetheless, we believe that our datasets, the processing pipeline and the evaluation framework can serve as a reference point for future evaluations of such embodied reasoning systems.

VII. Acknowledgement

This work was supported by the project Interactive Perception-Action-Learning for Modelling Objects (IPALM) (H2020–FET–ERA-NET Cofund–CHIST-ERA III / Technology Agency of the Czech Republic, EPSILON, no. TH05020001 / UK EPSRC EP/S032398/1). J.K.B. was supported by the European Regional Development Fund under project Robotics for Industry 4.0 (reg. no. *CZ*.02.1.01/0.0/0.0/15_003/0000470). Y.D. is supported by a RAEng Chair in Emerging Technologies.

REFERENCES

[1] M. Nazarczuk and K. Mikolajczyk, "V2A-Vision to Action: Learning robotic arm actions based on vision and language," in *Asian Conference on Computer Vision (ACCV)*, 2020.

[2] A. Das, S. Datta, G. Gkioxari, S. Lee, D. Parikh, and D. Batra, "Embodied question answering," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[3] L. Yu, X. Chen, G. Gkioxari, M. Bansal, T. L. Berg, and D. Batra, "Multi-target embodied question answering," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[4] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask R-CNN," in *International Conference on Computer Vision (ICCV)*, 2017.

[5] Z. Murez, T. van As, J. Bartolozzi, A. Sinha, V. Badrinarayanan, and A. Rabinovich, "Atlas: End-to-end 3D scene reconstruction from posed images," in *Computer Vision - ECCV*, 2020.

[6] T. Hodaň, D. Baráth, and J. Matas, "EPOS: Estimating 6D pose of objects with symmetries," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[7] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. M. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei, "Language models are few-shot learners," *CoRR*, 2020.

[8] D. Nyga, S. Roy, R. Paul, D. Park, M. Pomarlan, M. Beetz, and N. Roy, "Grounding robot plans from natural language instructions with incomplete world knowledge," in *Proceedings of Machine Learning Research*, vol. 87, 2018, pp. 714–723.

[9] E. Corona, A. Pumarola, G. Alenyà, F. Moreno-Noguer, and G. Rogez, "Ganhand: Predicting human grasp affordances in multi-object scenes," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[10] A. H. Qureshi, A. Simeonov, M. J. Bency, and M. C. Yip, "Motion planning networks," in *International Conference on Robotics and Automation (ICRA)*, 2019.

[11] G. Schwartz and K. Nishino, "Recognizing material properties from images," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 42, no. 8, pp. 1981–1995, 2020.

[12] D. Driess, O. Oguz, J. S. Ha, and M. Toussaint, "Deep visual heuristics: Learning feasibility of mixed-integer programs for manipulation planning," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2020.

[13] Z. Wang, C. R. Garrett, L. P. Kaelbling, and T. Lozano-Perez, "Active model learning and diverse action sampling for task and motion planning," in *International Conference on Intelligent Robots and Systems (IROS)*, 2018.

[14] M. Shridhar, J. Thomason, D. Gordon, Y. Bisk, W. Han, R. Mottaghi, L. Zettlemoyer, and D. Fox, "Alfred a benchmark for interpreting grounded instructions for everyday tasks," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[15] B. Calli, A. Singh, A. Walsman, S. Srinivasa, P. Abbeel, and A. M. Dollar, "The YCB object and model set: Towards common benchmarks for manipulation research," in *Proceedings of the 17th International Conference on Advanced Robotics (ICAR)*. Institute of Electrical and Electronics Engineers Inc., 2015, pp. 510–517.

[16] M. Nazarczuk and K. Mikolajczyk, "SHOP-VRB: A Visual Reasoning Benchmark for Object Perception," in *International Conference on Robotics and Automation (ICRA)*, 2020.

[17] J. Dzifcak, M. Scheutz, C. Baral, and P. Schermerhorn, "What to do and how to do it: translating natural language directives into temporal and dynamic logic representation for goal management and action execution," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2009.

[18] J. K. Behrens, K. Stepanova, R. Lange, and R. Skoviera, "Specifying dual-arm robot planning problems through natural language and demonstration," *IEEE Robotics and Automation Letters*, vol. 4, no. 3, pp. 2622–2629, 2019.

[19] C. Moulin-Frier, T. Fischer, M. Petit, G. Pointeau, J.-Y. Puigbo, U. Pattacini, S. C. Low, D. Camilleri, P. Nguyen, M. Hoffmann, H. J. Chang, M. Zambelli, A.-L. Mealier, A. Damianou, G. Metta, T. J. Prescott, Y. Demiris, P. F. Dominey, and P. F. Verschure, "DAC-h3: A proactive robot cognitive architecture to acquire and express knowledge about the world and the self," *IEEE Transactions on Cognitive and Developmental Systems*, vol. 10, no. 4, pp. 1005–1022, 2018.

[20] J. Roh, C. Paxton, A. Pronobis, A. Farhadi, and D. Fox, "Conditional driving from natural language instructions," in *3rd Conference on Robot Learning (CoRL)*, 2019.

[21] X. Wang, Q. Huang, A. Celikyilmaz, J. Gao, D. Shen, Y.-F. Wang, W. Y. Wang, and L. Zhang, "Reinforced cross-modal matching and self-supervised imitation learning for vision-language navigation," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[22] A. B. Vasudevan, D. Dai, and L. V. Gool, "Talk2nav: Long-range vision-and-language navigation with dual attention and spatial memory," *International Journal of Computer Vision*, 2020.

[23] M. Edmonds, F. Gao, H. Liu, X. Xie, S. Qi, B. Rothrock, , Y. Zhu, Y. N. Wu, H. Lu, and S.-C. Zhu, "A tale of two explanations: Enhancing human trust by explaining robot behavior," *Science Robotics*, vol. 4, no. 37, p. eaay4663, 2019.

[24] P. Anderson, Q. Wu, D. Teney, J. Bruce, M. Johnson, N. Sünderhauf, I. Reid, S. Gould, and A. van den Hengel, "Vision-and-language navigation: Interpreting visually-grounded navigation instructions in real environments," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[25] S. Chitta, I. Sucan, and S. Cousins, "Moveit! [ros topics]," *IEEE Robotics & Automation Magazine*, vol. 19, no. 1, p. 18–19, Mar 2012.

[26] Blender Online Community, *Blender - a 3D modelling and rendering package*, Blender Foundation, Stichting Blender Foundation, Amsterdam, 2018. [Online]. Available: http://www.blender.org

[27] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[28] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," in *Advances in Neural Information Processing Systems*, 2014.

[29] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, pp. 1735–1780, 4 1997.

[30] H. Liang, X. Ma, S. Li, M. Görner, S. Tang, B. Fang, F. Sun, and J. Zhang, "Pointnetgpd: Detecting grasp configurations from point sets," in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 3629–3635.

[31] E. Todorov, T. Erez, and Y. Tassa, "Mujoco: A physics engine for model-based control," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, Oct 2012, p. 5026–5033.

[32] S. Chitta, E. Marder-Eppstein, W. Meeussen, V. Pradeep, A. Rodríguez Tsouroukdissian, J. Bohren, D. Coleman, B. Magyar, G. Raiola, M. Lüdtke, and E. Fernández Perdomo, "ros_control: A generic and simple control framework for ROS," *The Journal of Open Source Software*, 2017.