Interpreting Manipulation Actions: From Language to Execution

Bao-Anh Dang-Vu, Oliver Porges, and Máximo A. Roa

Institute of Robotics and Mechatronics, German Aerospace Center (DLR)
82234 Wessling, Germany
{firstname.lastname}@dlr.de,
http://www.robotic.dlr.de

Abstract. Processing natural language instructions for execution of robotic tasks has been regarded as a means to make more intuitive the interaction with robots. This paper is focused on the applications of natural language processing in manipulation, specifically on the problem of recovering from the instruction the information missing for the manipulation planning, which has been traditionally assumed to be available for instance via pre-computed grasps or pre-labeled objects. The proposed approach includes a clustering process that discriminates areas on the object that can be used for different types of tasks (therefore providing valuable information for the grasp planning process), the extraction and consideration of task information and grasp constraints for solving the manipulation problem, and the use of an integrated grasp and motion planning that avoids relying on a predefined grasp database.

Keywords: manipulation, grasp planning, natural language processing

1 Introduction

Teaching and interacting with robots is shifting from being a highly specialized task requiring specific knowledge, to a more natural interaction that allows agnostic users to instruct the robot in the same way as they would instruct a human apprentice. Such capability requires the ability to process natural language (NL) instructions coming from a human. For instance, a simple instruction like "Bring me a glass of water" requires a decomposition into a set of simpler subtasks (look for the glass, pick it up, verify if it already contains water, go to the faucet, open the faucet, pour water onto the glass, close the faucet, move the glass to an inferred goal position), and in turn each one of the subtasks requires inferring the objects involved in the action (glass, faucet), and the required action (pick up, hold, open, move). Environmental constraints are tacitly embedded into the instructions; for instance, if the glass already contains water, then the instruction is simply translated into a pick and place command.

To successfully execute the tasks, the robot must be able to recognize the objects in the environment, and infer and understand the spatial relations between them. For instance, if the glass must be filled with water, the robot should recognize that it needs to go to the kitchen, a physical location in a home environment [10]. Physical spaces define then a subproblem of navigation, which can also be modified via spatial constraints as in "do not go to the toilet", or "stay away from the stairs". Prepositions embedded into the NL are often used to express these spatial relations, and have been considered either through predefined, hard-coded models of relational primitives, or through probabilistic inference of the spatial meaning that they enclose [14]. Typical spatial prepositions include, among others: to, from, along, across, through, toward, past, into, onto, out of, and via. Learning and probabilistic reasoning have been also used, for instance, for the grounding and semantic interpretation of phrases that lead to the definition of navigational paths for service robots [4], or for the creation of an extended vocabulary for tabletop manipulation scenarios [6].

Natural instruction of robotic manipulation problems has become an active research topic. Typical everyday activities for humans involve a large amount of inferred and possibly hidden knowledge about the way the activity should be performed, what objects are required to solve a specific task, the relation between them, and the affordances that each one of those objects allows [10]. Several works have tackled the grounding of NL commands to robot instructions for these cases. The concept of Probabilistic Robot Action Cores (PRAC) was introduced to encode action-specific knowledge in a probabilistic knowledge database, which helps to solve ambiguity and underspecification of the NL commands [10]. Inclusion of environment and task context and variation of language has also been considered by learning from example sequences performed in an online simulated game [9]. There, a learning model based on an energy function that encodes the validity of pre- and post-conditions, length of the instructions and consistency, was used to translate sequences of NL commands into sequences of instructions appropriate for the environment and task at hand. Learning the spatial semantics of manipulation actions can also be obtained through the analysis of the evolution of spatial relations between objects [15].

This paper is also focused on the problem of performing manipulation tasks with instructions given in natural language (Fig. 1). Different to previous works, our aim is to exploit the information contained in the NL expression to deduce the appropriate constraints for simple skills, in this case grasping and manipulating an object. For instance, if the object is an electric driller, a user could specify a pick and place action on the object, which only requires a stable grasp on it, or could suggest drilling a hole on some other material, which necessarily translates into constraints for grasping and activating the object to perform the intended task. Therefore, given a NL expression we parse the sentence and exploit its syntactical structure to determine actions and semantic relations between the objects. Action-specific knowledge is extracted from the instruction, not only to deduce the current location and goal region for the manipulated object, but also to understand how the object should be grasped, or if there are special restrictions on the way it should be manipulated. Naturally, this knowledge must be complemented with a suitable understanding of the object-specific affordances. A semantic clustering method is used to obtain regions that could be employed

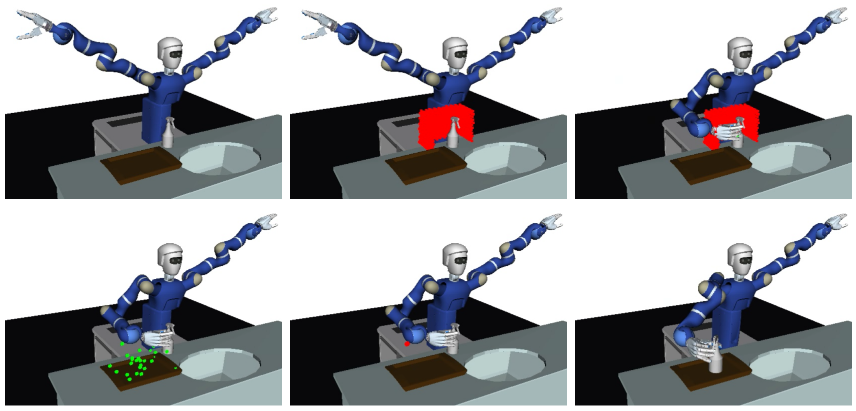

Fig. 1. Sequence of planned actions from the instruction "put the spray on top of the tray," using the DLR robot SpaceJustin: a) initial configuration; b) potential approach directions, obtained with a capability map; c) initial grasp: d) sampling the goal location; e) object being transported; f) object in final location.

with different purposes on the object. The desired goal, the task constraints and the grasp restrictions are considered in an integrated grasp and motion planner that derives feasible trajectories for performing the manipulation action.

After this introduction, the paper is organized as follows. Section 2 describes the overall architecture of the planning system. Section 3 presents the framework used to analyze the NL instruction and extract constraints for the instantiation of the manipulation task. Section 4 explains the inclusion of these constraints on the motion planner. Finally, Section 5 concludes the paper.

2 Overall concept

A simple human-to-human instruction, like "cut the bread", involves hidden knowledge that a service robot should infer: *cut* is an action that requires an agent object, a knife, which is a tool and has a proper way to be used, and *bread* is a passive object, the object that is acted upon to perform the action. The goal of our system is to interpret the meaning coming from an input sentence in order to send appropriate commands to the robot to perform the grasping or manipulation skill. First, the natural language text is processed using a statistical parser. The syntactic structure is exploited in order to determine action verbs and semantic relations between the objects. Possibly hidden relations should be inferred also at this level. Then, the action-specific knowledge is deduced, i.e. what to do to which objects in a particular situation. The passive object that is acted upon is deduced by analyzing the prepositional relations (with, from, to, into) that modify the corresponding action verb. The action verb describes a range of tasks that we simplify into three categories: pick and place tasks (direct manipulation), tool usage, or device usage. We make here a subtle distinction between tool and device by considering that a device contains a trigger that should be pressed/moved to generate the intended action (e.g. an electric driller), while the tool is just an object with some affordance that allows its usage to perform some particular task (e.g. a knife). This distinction is critical for the grasp planning process, as a device requires an active finger capable of triggering the desired action while the other fingers grab the device. The objects, nouns and prepositions in the NL instruction help also to decide the spatial relations, i.e., to identify an initial and final pose for an object. For instance, the sentence "Take the mug that is behind the plate and put it on the tray" provides some hints on where the mug can be found, indicates the goal position, and defines a pick and place action. A vision module (not explained in this paper) identifies the required objects and estimates its location, and provides information to infer possible goal positions for the grasping or manipulation action.

To complement the knowledge coming from the NL instruction, the system is supported by an object database that includes the semantic knowledge of the objects. The functional parts of the objects could be manually labeled, or can also be obtained in a more systematic way through an active clustering process, as proposed here. In any case, one object in the database, for instance a spoon, has a pre-stored geometrical interpretation of its parts: a handle, where the object should be grasped, and a bowl, the part required to perform some action. Also, the database includes predefined tasks or action verbs associated to the object: pick and place, serve, mix.

With all the previous elements in consideration, a planner is executed to create a grasp and a path –or a sequence of paths– for the arm and end effector to successfully accomplish the task. The task specification defines grasp constraints on the object, and the initial and final location defines the starting and goal configuration for the path planner. To simultaneously consider these restrictions, an integrated grasp and motion planner decides the best grasp on the object and the collision-free path that allows the successful execution of the task [5].

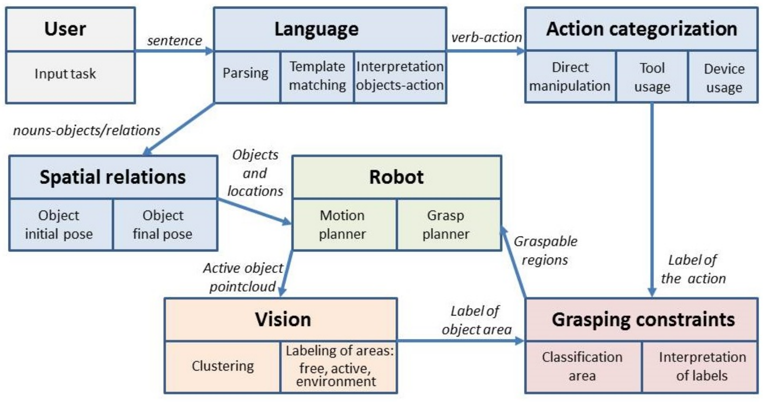

Fig. 2. Architecture of the system.

The architecture of the overall system is presented in Fig. 2. This paper is focused on the NL processing system, and is presented in two parts: 1) Extraction of constraints: spatial relations, current and desired object location, role of the object, desired task (Section 3), and 2) use of the action-specific knowledge to plan the motion: constrained areas and specifications of required contact points for grasping, and integrated grasp and motion planning (Section 4). Note that the generation of long sequences of actions, for instance for making a pancake [1], considerations on the performance of the NL processing like generality, templating and instantiation of actions or parsing success, and an extensive evaluation of the system performance are out of the scope of the paper; the paper is devoted to the description of the proposed architecture for extracting the task-related knowledge from the NL instruction, in order to provide parameters for the generation of the plan required for executing the manipulation task.

3 Natural language processing: from task specification to description of constraints

Processing the natural language instruction not only provides the definition of the task, but also includes important clues on the intended task, mainly:

– the role of the objects: an object can be active, like a tool/device, or passive, i.e., modified by the action of some other object. Note that a given object can be passive or active, depending on the provided NL instruction.
– information on the spatial relations: defines for instance possible reference locations for the objects, or an intended goal area for a pick and place task.

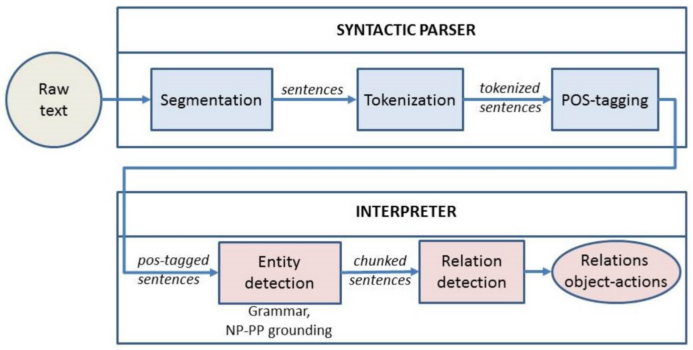

Fig. 3. Language processing approach.

To perform the NL processing we use the Natural Language Toolkit, NLTK [3], a Python-based suite of libraries and programs for processing free text. The general approach for the language processing is summarized in Fig. 3. The syntactic parser is used directly from NLTK. However, since NLTK does not include a semantic analyzer (interpreter), we developed our own semantic module, specifically suited for manipulation instructions.

3.1 Language processing: syntactic parsing

To correctly interpret a command such as "Put the bottle on the tray" or "Use the screwdriver", the system must ground and interpret the sentence and figure out the relations and states of the objects. The planning system is managed by an interface that allows the user to type free sentences (Fig. 2). The input is then a sentence in a free-form natural language, which can contain information about the objects through their names or properties (e.g., "the bottle", or "the red bottle behind the plate"), and description of their spatial relations with respect to other objects (e.g., "to the left of"). The syntactic parser module outputs the interpretation of the command as the action type (verb), the identity of the objects (nouns), and the spatial location (prepositions). The prepositions considered in this work for the grasping and manipulation skills are: above, behind, below, close to, far from, in front of, inside of, on, to the left of, to the right of, and under. The sentence is segmented and tokenized, i.e. divided in words, punctuation signs, etc., and POS (Part-Of-Speech)-tagged, i.e. tokens are marked with their corresponding word type based on the token itself and the context of the token.

3.2 Grammar: language grounding

The arbitrary structure of the natural language is reduced to a structure based on clausal decomposition. The clause $\mathcal{C}$ is defined by a tuple $\mathcal{C} = (a, [\text{role}], r)$ containing the action a described by the verb, the set [role] of *object-role* that defines the object to move as active and the referenced object (if present) as passive, and a relationship r between objects, deduced from the spatial relations (e.g., "left", "from").

In order to parse and describe the structure of the sentences, we use a grammar $\mathcal{G}$ that identifies the verb (V), noun (N), preposition (PP), noun-phrase (NP $\rightarrow$ N PP), and spatial relation (SP $\rightarrow$ NP PP). For example, for the sentence:

$$\underbrace{\text{“Put the bottle on the left of tray”}}_{\mathcal{C}\,=\,(a=\text{put},\ [\text{role}]=\{\text{bottle:active, tray:passive}\},\ r=\text{“on the left”})}$$

the use of the grammar generates the tree shown in Fig. 4.

The parse-tree is analyzed to find nodes of action-verb type that form a sentence, and each sentence is used to find a clause. The action-verb type retrieved by the clause can have multiple meanings in different contexts. For example, the action-verb "pick" has the same meaning as "take", and both belong to the same group of actions that require direct manipulation of the object (pick and place). Therefore, we perform a disambiguation step in order to classify each action-verb into specific categories, similar to the procedure of [16]. To

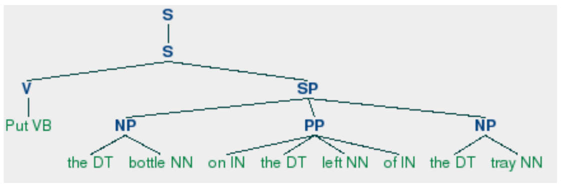

Fig. 4. Parsing: "Put the bottle on the left of tray".

understand the different meanings, we reduce the dimensionality of the set of action verbs by semantic clustering. We divide the actions into two categories: "direct manipulation" such as pick, hold, place, where the goal is to pick and move objects directly, and "tool/device use" such as screw or cut (e.g. non pick and place), where the object must enter in contact with the environment to perform some action.

The meaning of each action-verb is obtained using the corpus reader WordNet [8] that provides cognitive synonyms, or synsets, of a word. The synsets of "pick" are generated in this way, and the synsets for tool/device use are manually entered. During execution, the retrieved action-verb is compared to the list of synsets in order to classify the action into one of the two categories already defined.

3.3 Spatial relations

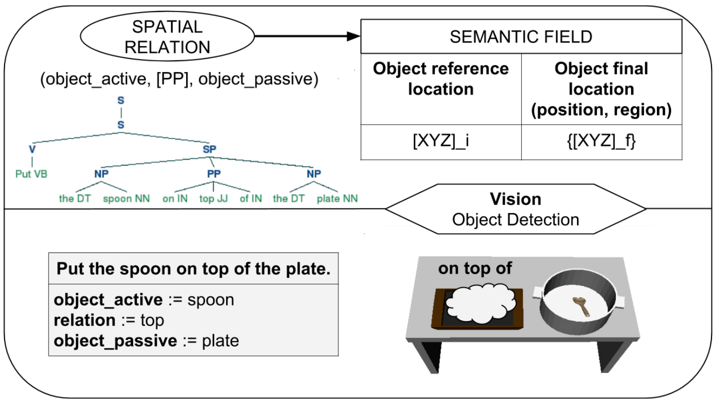

Fig. 5. Interpretation of spatial relations.

The deduction of spatial relations depends on the vision module, to consider the location of objects from the point of view of the robot, and on the prepositions present in the sentence, to interpret the spatial relations between the objects. The spatial relations r are described by prepositions such as "to the left of" or "near to". To interpret those relations in the environment, we use a semantic field model, sketched in Fig. 5.

The semantic field of a spatial relation is represented as a multivariate Gaussian distribution over 3D points that fall into the described area of the preposition, relative to the passive object, and assigns weight values to the points depending on the meaning of the preposition. For example, to reason about some active object being "to the left" of a passive object (a tray in the example of Fig. 6), the spatial region to the left of the passive object is considered (which depends on the current location of the robot), and then we compute the multivariate Gaussian where its mean is defined by the extents of the passive object.

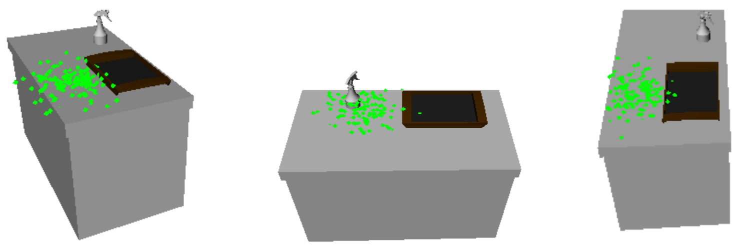

Fig. 6. Spatial language interpretation with semantic field depending on the point of view, for the instruction "Put the bottle to the left of the tray". The robot is assumed to be located perpendicular to the page.

4 Planning: from constraints to semantic grasps

The actions and spatial relations found from the NL define the constraints for the grasp and motion planning problem. The approach is supported by an object database that contains information on the functionality. Previous works dealing with the addition of such functional information to object databases rely on manual input from a human [7]. Here, this process is replaced by a clustering and annotation process that simplifies the generation of differentiated areas on the objects. This functional information and the task information coming from the NL processing are used to generate desirable areas or constraints for grasping the object. Finally, these constraints, along with the initial location of the object and the goal region defined by the sentence, are used as input to an integrated grasp and motion planner that generates a feasible trajectory for the robot.

4.1 Semantic clustering and annotations of objects

For utilizing an object, a robot must be aware of the affordances of the object, and the implications that they have in terms of restrictions on the object. For instance, a glass can be grasped in different ways, but if the glass is needed to pour a liquid into it, then the top part of the glass must be free to allow the pouring action. More complex objects are classified into the categories of tools/devices; these objects are meant to actively perform some action on another object or the environment. The semantic difference considered here to define a device is that it requires the actuation of a finger on some mobile part (button, lever, trigger) to initiate its expected action. Examples of devices are a driller or an electric screwdriver, while knives or pencils would be tools.

Natural language instructions, originally meant for humans, often do not explicitly specify the part of the object that should be considered for using the tool. For instance, the instruction "Use the spray bottle to clean the table" does not indicate the location of the spray trigger. Thus, we identify the functions of the object through symbolic attributes defined with three labels:

- Action area $\mathcal{A}$: area that the hand should interact with in order to perform an action.
- Environment area $\mathcal{E}$: area meant for the interaction with the environment, which cannot be touched or blocked (forbidden area).
- Free area $\mathcal{F}$: area where no constraints are applied, and therefore is free to be used by the gripper/multifingered hand to grasp the object.

For example, a driller is defined with three different parts: a trigger that must be touched in order to activate it, a main body that can be grasped, and a forbidden region that interacts with a screw (Fig. 7). For an object described with a pointcloud, each point belongs to one of the described categories, and the whole object $\mathcal{O}$ is the union of the three sets: $\mathcal{O} = \mathcal{A} \bigcup \mathcal{E} \bigcup \mathcal{F}$.

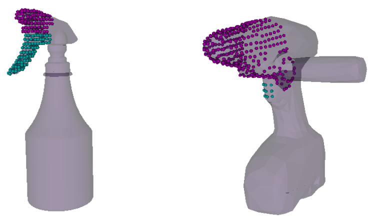
Fig. 7. Semantic clustering of a spray bottle and a driller: the action area $\mathcal{A}$ is displayed as blue dots, the environment area $\mathcal{E}$ as purple dots, and the remaining gray zone corresponds to the free area $\mathcal{F}$.

To achieve a clustering of the object, we use the Locally Convex Connected Patches (LCCP) method [13], which gives a rough clustering that approximately conforms to object boundaries. The clustering can be further refined in order to have the three distinct parts. The annotation of the object is then validated by a user; this validation is required only once for each object in the database. Methods to make this process in a completely autonomous way are currently under investigation.

4.2 Semantic grasps

The generation of the sets defined in the previous section helps to identify the graspable areas on the object, required for the grasp planner. The grasp planner used in this work is based on the concept of reachable Independent Contact Regions (rICR), i.e. regions (patches) that are reachable for the current hand pose and that guarantee a force closure grasp when each finger is (independently) located inside its corresponding contact region on the object surface [12].

The kinematics of the hand is considered by (offline) computing the reachable workspace for each one of the fingers. First, reachable points on the object surface are obtained by computing the collision points between the pointcloud of the object and the workspace for each finger. This leads to the reachable regions $\mathcal{R}$:

$$\mathcal{R} = \{\{xyz\}_{i=1,\cdots,\text{nFingers}}\}$$

Then, the labeled areas on the object (action, free and environment area) are used to obtain a subset of the reachable points that satisfies the constraints of the task. If a is a pick and place task, the reachable points in $\mathcal{F}$ can be used for grasping

$$\mathcal{G}_1 = \Big\{\{\mathcal{R}\bigcap\mathcal{F}\} - \{\mathcal{A}\bigcup\mathcal{E}\}\Big\}$$

If a is a task using a tool, the points used for grasping must include also points from $\mathcal{A}$, to be able to effectively use the object

$$\mathcal{G}_2 = \Big\{\{\mathcal{R}\bigcap\{\mathcal{F}\bigcup\mathcal{A}\}\} - \mathcal{E}\Big\}$$

These graspable areas $\mathcal{G}$ are the ones considered in the grasp planning process, as illustrated in Fig. 8 for a spray bottle considering a device usage or a simple pick and place task.

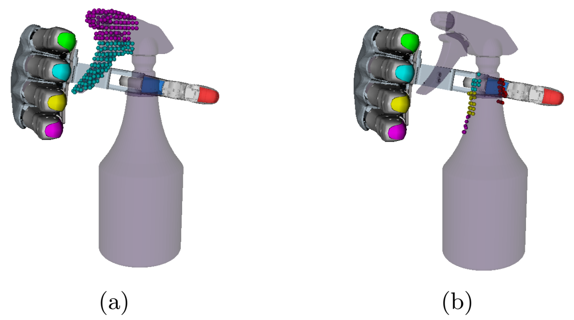

Fig. 8. Graspable regions for different tasks: a) The use of the spray bottle requires at least one finger to be located on the action area $\mathcal{A}$ (in blue), while the environment area $\mathcal{E}$ (in purple) should be free; b) the action area $\mathcal{A}$ is removed from the reachable regions, thus creating a graspable area $\mathcal{G}_2$ for a pick and place task, that avoids possible activation of the device while it is being transported.

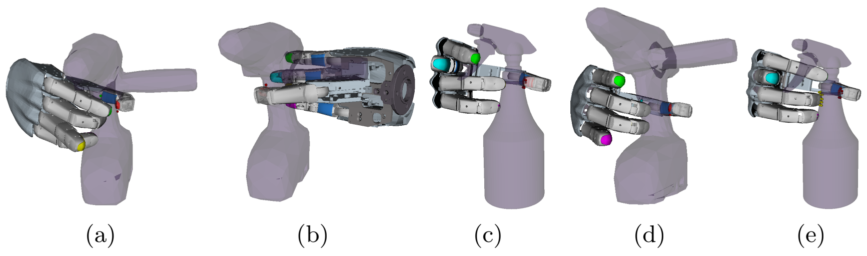

Fig. 9. Constrained grasps for action (a, b and c), and for pick and place (d and e).

Fig. 9 illustrates different grasps for a driller and a spray bottle, for use of the device and for pick and place tasks. Note that qualitatively different grasps can be obtained for using the driller, and still they meet the constraint of having one finger on the action button. In the examples of grasps for a pick and place task, the action area is avoided even though the hand is in its vicinity.

4.3 Integrated grasp and motion planner

Traditionally, grasp and arm motion planning are considered as separate tasks. This approach generally relies on a set of precomputed grasps (grasp database) for finding a grasp suitable for the current situation, which greatly limits the number of grasp possibilities that can be tried, i.e. there is a low adaptability to the environment as no new grasps can be explored, even if they mean only a slight change of pose with respect to one of the predefined grasps. The feasibility of the grasps is evaluated for the given scenario, and only grasps that have a corresponding inverse kinematics (IK) solution for the arm are considered in later stages. One feasible grasp defines one goal configuration for the robot. Then, given the initial and final arm configuration, a collision-free path for the arm is searched using some path planning method. If no path is found a new grasp is chosen, until a path is obtained or until the complete database has been explored and no solution is found [2].

A better adaptability is obtained when using an integrated grasp and motion planner that only requires the initial configuration of the arm and the pose of the target object to simultaneously plan a good hand pose and arm trajectory to grasp the object [5]. The planner relies on the computation of independent contact regions to look for the best possible grasp; the planner is here adapted so that the obtained region for each finger is a subset of the actual graspable area $\mathcal{G}$. The potential grasp poses are obtained by using a capability map, i.e. an offline computed representation of the reachable and dexterous workspace for the robotic manipulator [11]. Once a goal grasp has been identified, the corresponding collision-free motion for the arm and hand is obtained by using bidirectional RRTs (Rapidly-exploring Random Trees). The integrated planner can be sequentially called as required; for instance, for pick and place tasks an initial plan is required to approach and grasp the object, and a second plan is used to move the object from the initial to the final configuration.

5 Final comments

This paper presented a system to process a natural language instruction, with the main focus being how to retrieve the information required for instantiating and executing a grasp or manipulation skill. Given a task description in natural language, the system grounds the sentences in order to find spatial relations between objects, and to discover the meaning of the task itself. The system relies on an associated object database that contains information on the possible usage (action verbs) and the geometrical regions of the object, which are obtained through an automatic clustering process whose annotations are later validated by a user. This allows the generation of grasp constraints that are inherent to the object to be used, and therefore independent of the end effector of the particular application. The semantic knowledge of the object and the task allows the reduction of the search space explored during the grasp planning process. A vision module, not considered in this paper, additionally provides information on the location of objects and goal locations, as specified in the NL instruction. Once the manipulation skill is completely specified, including identification and role of the involved objects, desired usage of the object and current and goal locations, an integrated grasp and motion planner generates the collision-free trajectory required for the skill execution.

The described pipeline was implemented on top of OpenRAVE , and preliminary tests of the approach in simulation were successful for different instructions for tabletop scenarios. Some examples of the tested instructions are: “take the spray”, “clean with the spray”, “put the spray on top of the tray”, “use the drill”, “grab the drill”, “put the drill to the right of the spray”. Fig. 1 shows a sequence of snapshots identifying different steps in the generation of the plan for the instruction “put the spray on top of the tray”, executed for the DLR robot SpaceJustin, using two five-fingered DLR/HIT hand II as end effectors. The planning time varies from a few seconds to about a minute for more complex problems. The main factor that influences the planning time is not the NL processing module, but the relative location and reachability of the required objects. Next steps for the development of this system include the expansion of the vocabulary that the NL processor considers, the development of an automatic clustering algorithm that avoids the need for user intervention to verify the annotations generated by the system, an extensive and quantitative study of the performance of the system, and the integration of the system and evaluation of the success rate on a real robotic system.

Acknowledgment. The research leading to these results has received partial funding through the project DPI2013-40882-P.

References

1. Beetz, M., Kresse, U.K.I., Maldonado, A., Mosenlechner, L., Pangercic, D., Ruhr, T., Tenorth, M.: Robotic roommates making pancakes. In: Proc. IEEE-RAS Int. Conf. on Humanoid Robots. pp. 529–536 (2011)
2. Berenson, D., Diankov, R., Nishikawi, K., Kagami, S., Kuffner, J.: Grasp planning in complex scenes. In: Proc. IEEE-RAS Int. Conf. on Humanoid Robots. pp. 42–48 (2007)
3. Bird, S., Loper, E.: NLTK: the Natural Language Toolkit. In: Proc. ACL (Association for Computational Linguistics) on Interactive presentation sessions (2004)
4. Fasola, J., Mataric, M.J.: Using semantic fields to model dynamic spatial relations in a robot architecture for natural language instruction of service robots. In: Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems. pp. 143–150 (2013)
5. Fontanals, J., Dang-Vu, B., Porges, O., Rosell, J., Roa, M.A.: Integrated grasp and motion planning using independent contact regions. In: Proc. IEEE-RAS Int. Conf. on Humanoid Robots. pp. 887–893 (2014)
6. Guadarrama, S., Riano, L., Golland, D., Gouhring, D., Jia, Y., Klein, D., Abbeel, P., Darrell, T.: Grounding spatial relations for human-robot interaction. In: Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems. pp. 1640–1647 (2013)
7. Leidner, D., Borst, C., Hirzinger, G.: Things are made for what they are: Solving manipulation tasks by using functional object classes. In: Proc. IEEE-RAS Int. Conf. on Humanoid Robots. pp. 429–435 (2012)
8. Miller, G.A.: Wordnet: a lexical database for English. Communications of the ACM 38(11), 39–41 (1995)
9. Misra, D., Sung, J., Lee, K., Saxena, A.: Tell me Dave: Context-sensitive grounding of natural language to mobile manipulation instructions. In: Proc. Robotics: Science and Systems, RSS (2014)
10. Nyga, D., Beetz, M.: Everything robots always wanted to know about housework (but were afraid to ask). In: Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems. pp. 243–250 (2012)
11. Porges, O., Stouraitis, T., Borst, C., Roa, M.A.: Reachability and capability analysis for manipulation tasks. In: Armada, M., Sanfeliu, A., Ferre, M. (eds.) ROBOT2013: First Iberian Robotics Conference, pp. 703–718. Advances in Intelligent Systems and Computing 253, Springer (2014)
12. Roa, M.A., Hertkorn, K., Borst, C., Hirzinger, G.: Reachable independent contact regions for precision grasps. In: Proc. IEEE Int. Conf. on Robotics and Automation. pp. 5337–5343 (2011)
13. Stein, S.C., Schoeler, M., Papon, J., Worgotter, F.: Object partitioning using local convexity. In: Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR). pp. 304–311 (2014)
14. Tellex, S., Kollar, T., Dickerson, S., Walter, M.R., Banerjee, A., Teller, S., Roy, N.: Understanding natural language commands for robotic navigation and mobile manipulation. In: Proc. Nat. Conf. on Artificial Intelligence - AAAI (2011)
15. Zampogiannis, K., Yang, Y., Fermueller, C., Aloimonos, Y.: Learning the spatial semantics of manipulation actions through preposition grounding. In: Proc. IEEE Int. Conf. on Robotics and Automation. pp. 1389–1396 (2015)
16. Zoliner, R., Pardowitz, M., Knoop, S., Dillmann, R.: Towards cognitive robots: Building hierarchical task representations of manipulations from human demonstration. In: Proc. IEEE Int. Conf. on Robotics and Automation. pp. 1535–1540 (2005)