Singapore Management University

Institutional Knowledge at Singapore Management University

Research Collection School Of Information Systems

School of Information Systems

8-2019

# Language and robotics: Complex sentence understanding

Seng-Beng HO

Zhaoxia WANG
*Singapore Management University*, zxwang@smu.edu.sg

Follow this and additional works at: https://ink.library.smu.edu.sg/sis_research

Part of the Artificial Intelligence and Robotics Commons

## Citation

1

This Conference Proceeding Article is brought to you for free and open access by the School of Information Systems at Institutional Knowledge at Singapore Management University. It has been accepted for inclusion in Research Collection School Of Information Systems by an authorized administrator of Institutional Knowledge at Singapore Management University. For more information, please email cherylds@smu.edu.sg.

# Language and Robotics: Complex Sentence Understanding

Seng-Beng Ho[1,2](&) and Zhaoxia Wang[1,3]

[1] Institute for High Performance Computing, A*STAR, Singapore, Singapore
hosengbeng@gmail.com, wangz.ihpc@gmail.com

[2] AI Programme A*STAR, Singapore, Singapore

[3] Nanjing University of Information Science and Technology, Nanjing, China

Published in Intelligent Robotics and Applications: 12th International Conference, ICIRA 2019, Shenyang, China, August 8-11: Proceedings, pp. 641-654. LNCS 11745.
DOI: 10.1007/978-3-030-27529-7_54

Abstract

Existing robotic systems can take actions based on natural language commands but they tend to be only simple commands. On the other hand, in the domain of Natural Language Processing (NLP), complex sentences are processed, but this NLP domain does not make close contact with robotics. The beginning of computer processing of natural language, when traced back to a system such as Winograd's SHRUDLU, conceived in 1973, actually aimed to address the issues of Natural Language Understanding (NLU) of relatively complex sentences by a robotic system which in turn takes actions accordingly based on the natural language input. NLU, in the robotic context, thus constitutes taking the correct actions from language instructions. This paper explores the use of cognitive linguistic constructs as well as other constructs such as spatial relationship constructs to configure an NLU system for translating complex natural language instructions into actions to be taken by a robot. This research work illustrates that two important steps are necessary: the first step is to translate a language-dependent surface sentential structure into a language independent deep-level predicate representation, and then the next step is to translate the predicate representation into grounded real-world references and constructs that enable a robot to carry out the language instructions accordingly.

Keywords: Complex sentence understanding, Grounding, Language and robotics, Natural language understanding, Predicate meaning representation, Predicate to referent grounding, Robotics, Semantic grounding

## 1 Introduction

This paper explores the relationship between language and robotics as well as the representational and computational devices needed for the understanding of complex natural language instructions. In the past decade or more, the research in Artificial Intelligence (AI) has splintered into many different "sub-areas" such as Natural Language Processing (NLP), Computer Vision (CV), robotics, machine learning, etc. In some cases, there are stronger connections, such as between machine learning and the other fields, because machine learning has to be applied to some problems (NLP, CV, etc.). There are also strong connections between CV and robotics. However, even though robotics is connected to NLP in the sense that some robotic systems do employ

NLP when it is desired that robots respond to natural language commands, these commands tend to be relatively simple [1–3]. On the other hand, NLP people largely proceed with their research as though robotics does not exist [4–7]. But there is actually a very strong connection between the two aspects of intelligent systems. One of the very first NLU systems, Winograd's SHRUDLU, conceived in 1973, consists of a toy-world in which a robotic arm is made to carry out actions based on natural language commands [8]. This work illustrates the inseparability between language and robotics: through carrying out the respective actions correctly, the system demonstrate that it really "understands" the natural language input, and this distinguishes NLU from "merely" NLP. And from a robotic system's point of view, it would naturally benefit from being able to carry out actions based on natural language commands.

This paper first reviews some of the representative past work which demonstrates the close connections between language and robotics, and then proceeds to describe a method to enable robots to receive complex natural language commands and carry out the respective actions accordingly. This research will contribute to human-like next generation AI.

## 2 Review of Relevant Work

Recent work in NLP focuses more on the "processing" aspect of natural language rather than addresses the issue of "understanding" [4–7]. E.g., a machine translation could output a string of words given another string of words in another language, but it does not really "understand" what they mean, nor perhaps does it really matter as far as the task is concerned. However, there has also not been many attempts in trying to define what "understanding" really entails. Ho and Wang [9] attempts to clarify what "understanding" really entails and in the subsequent discussions we will demonstrate that the process discussed not only clarifies what "understanding" really entails, it also provides an NLU framework for robots to receive complex natural language commands and act on them accordingly.

One of the earliest works relevant to NLU is Winograd's SHRUDLU system, conceived in 1973 [8]. Figure 1 shows a "toy-world" used in the system. The system basically involves a user typing in a series of commands and the system would answer some questions, or through a robotic arm, would carry out a series of actions accordingly. Thus, from the outset, language and robotics are intertwined in this paradigm of NLU. It is almost like Winograd is declaring "understanding is understanding how to act."

Issues of ambiguous reference are resolved through some built-in heuristics as well as clarifications requested by the system to the human user giving the commands. For example, as shown in the figure, it uses heuristics to determine what "it" refers to in one sentence and which block is being referred to in another sentence.

SHRUDLU suffered from the issue of scalability. At that time (the 70's) there was no machine learning or computer vision to assist in learning rules, say, for mapping between words and their referents in the real world, and hence the knowledge used had to be hand-coded. However, today there are systems that can achieve both the learning and word referents as well as the grammatical rules is possible concurrently [10].

>> Pick up a big grey block.
➢ Ok.

>> Find a block that is taller than the one you are holding and put it on the cylinder.
> BY “IT,” I ASSUME YOU MEAN THE BLOCK WHICH IS TALLER THAN THE ONE I AM HOLDING.
Ok.

>> What is the block supported by?
➢ THE CYLINDER.
(If this question was asked earlier, it would have responded “I DON’T UNDERSTAND WHICH BOX YOU MEAN”)

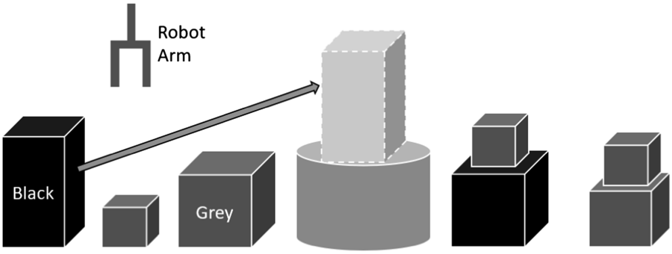

**Fig. 1.** Winograd’s SHRUDLU system [8].

Another piece of natural language related research that came from the 1970’s that is relevant to robotics and vision is Schank and Abelson’s work on Scripts [11]. Scripts basically consist of Causal-Spatio-Temporal (CST) descriptions of an event, such as the happenings in a restaurant, that allow complex question-answering exchanges to take place. Suppose someone says something like “I went to the Restaurant yesterday. I didn’t leave a tip,” the Restaurant Script may allow questions such as this to be responded to by a system: “Was the service bad?” as this piece of knowledge is encoded in the script’s processes describing what happen and why in a typical restaurant. If a Restaurant is simply defined as “an establishment where food can be purchase,” as in a typical dictionary definition, “deep understanding” reflected by this sophisticated linguistic response cannot be achieved.

In the 70’s these Scripts were hand-coded but recently it has been shown that CST graphs are learnable through observing the environment directly [12, 13].

There has been a number of linguistic paradigms [14] that attempt to formulate semantics – i.e., the meaning of meaning, or what meaning entails. Among all these paradigms, “cognitive linguistics” is most suited for providing useful representational constructs for AI and robotics [15].

Figure 2 shows some examples of representational constructs proposed in cognitive linguistics to describe spatial and temporal relationships. Figure 2(a) depicts the “above” and “below” relationships. It can be shown that as far as the objects involved are concerned (Object1 and Object2), the spatial arrangement between them remains the same in both descriptions. However, in a sentence such as “Object1 is *above* Object2,” cognitive linguistics identifies Object1 as the Trajector (TR) and Object2 as the Landmark (LM). TR is like a “focus” of the sentence and LM is like a “reference.” Therefore, should the positions of TR and LM be interchanged, then the language description becomes “Object2 is *below* Object1.” Thus *above* and *below* are in a complementary relationship, depending on the TR and LM.

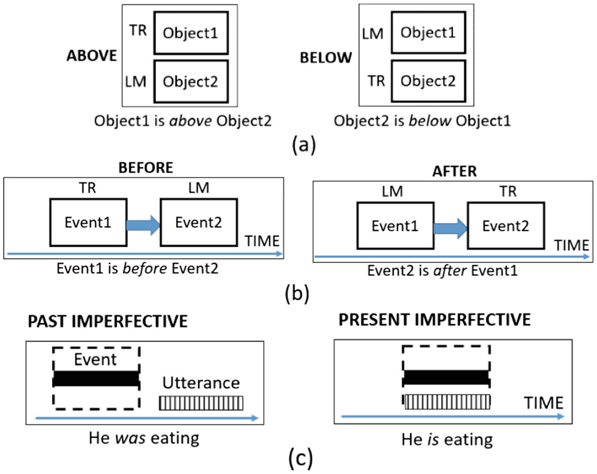

**Fig. 2.** Cognitive linguistics representations of (a) *above* and *below*; (b) *before* and *after*; (c) past and present imperfective [15].

Figure 2(b) depicts a similar situation, but in the temporal domain – the “before” and “after” relationships. The two events, Event1 and Event2, bear the same relative relationship in time in both situations, but there are two possible sentential descriptions arising from it, depending on which is the TR and LM.

Figure 2(c) depicts the meaning of the present imperfective and the past imperfective tenses in a temporal picture. The corresponding tense is used depending on the temporal relationship between the utterer of the sentence and the event itself.

In the following, we will leverage some of these cognitive linguistic representations for the use of robotic language understanding.

## 3 Language and Robotics: Complex Sentence Understanding

In this section, we elucidate the intimate connections between language and robotics, and show how some of the concepts reviewed above can be integrated to bear on the problems of complex sentence understanding. Due to the limitation of space, the discussions here are not meant to be exhaustive but merely illustrative.

### 3.1 Basic Instructions

We consider two kinds of instructions given to a robot:

AFFORDANCE:
Use OBJECT1 for TASK1 (AFFORDANCE)

TEMPORAL SEQUENCE:
Do TASK1 *then (begin to do*) TASK2
*After* you *have done* TASK1 *then* do TASK2
Do TASK1 *until* TASK2/EVENT1 *begins*
*While* you *are doing* TASK1, do TASK2

The first kind of instructions is to inform the robot what entity can be used to achieve a certain goal, i.e., what can *afford* what. (E.g., "use the screw driver to screw the screw" – a screw driver affords screwing of screws.) The second kind of instructions is to inform the robot what *temporal steps* it should take. Here we are assuming that words like "use," "for," "do," "you," etc. have built-in procedural meanings (i.e., procedures are used to implement them) and we do not explore how they may be represented explicitly such as in the cognitive linguistics examples for certain concepts in Fig. 2.

For the words in italics, such as "*then*," "*after*," "*while*," "*have done*," "*are doing*," "*until*," "*begin*," they are grounded in the corresponding cognitive linguistic constructs such as shown in Fig. 2.

Note that of course when a certain instruction such as "use a screw driver to screw the screw" is given to a totally naïve "infant" robotic system, the system still needs to work out, through a problem solving process, *how* exactly to use the screw driver. But this instruction at least helps to cut down the search space tremendously, otherwise thousands of objects may have to be tried.

Of course, the system can recurse: it can ask further questions and receive further language instructions: "How do I use that screw driver?" The answer(s) may be:

(1) "*First*, you pick-up that screw driver, and *then* you position the screw driver such that its long axis is aligned with the long axis of the screw, and *then* you insert the tip of the screw driver into the grooves on the top part of the screw, and *then* you rotate the screw driver around its long axis in a clockwise direction…"

The system may further ask, "How do I pick up the screw driver?" And the answer may be:

(2) "*First*, you position your hand directly above the screw driver's handle area, *then* you rotate your hand until the main axis of the gap between your fingers is aligned with the long axis of the screw driver's handle, *then* you lower your hand until these two axes coincide, *then* you close your fingers onto the handle, …"

If these complex language instructions can be "understood," the appropriate actions can be carried out. This will be explored in the subsequent sections. Figure 3 shows how the various sentences above can be represented in an explicit temporal representation through a syntactic transformation process.

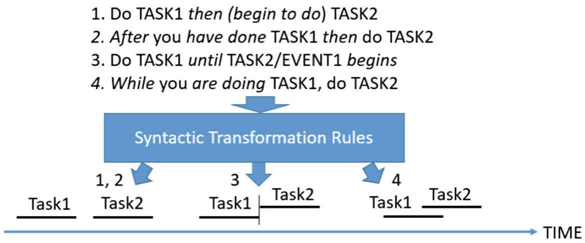
1. Do TASK1 *then (begin to do)* TASK2
2. *After* you *have done* TASK1 *then* do TASK2
3. Do TASK1 *until* TASK2/EVENT1 *begins*
4. *While* you *are doing* TASK1, do TASK2
Syntactic Transformation Rules
1, 2
3
4
Task1
Task2
Task1
Task2
Task1
Task2
TIME

**Fig. 3.** Syntactic transformation converts the sentences 1, 2, 3, and 4 into the corresponding temporal structures.

There is ambiguity inherent in the co-temporal instruction *while* and the temporal instruction *after…then*. The two tasks in a *while* construct may begin at the same time or at slightly different times, though they must overlap temporally, and the two tasks in the *after…then* construct may be separated by any amount of time. The robotic system may (i) decide on the time interval based on some earlier learned typical values; (ii) decide on the time interval based on the knowledge about the tasks at hand; and (iii) ask for further instructions.

For (iii), further language processing is needed, and the instructions may be:

Do TASK2 *while* doing TASK1, *begin* TASK2 **1 minute** *after starting* TASK1
Do TASK2 *after* TASK1, *begin* TASK2 **1 minute** *after stopping* TASK1

The new words introduced here and their corresponding progressive versions are "*start*" and "*stop*". *Start* has the same meaning in this context as *begin*.

For the case of "Do TASK1 *until* TASK2/EVENT1 *begins*," it implies that TASK1 could have continued but its termination is effected by the beginning of TASKS2 or another EVENT1.

## 3.2 Complex Instructions

There are a number of basic constructs that are needed before the complex instructions (1) and (2) discussed above can be "understood" and they are described as follows.

**Basic Constructs Needed**

***Objects and Parts***

Figure 4 shows an arrow-shaped object. For a longish and asymmetrical object such as this, it is typical for humans to identify, say by pointing or through the use of words, a "front" and a "back" parts. These parts could also be identified by their shapes – the "rectangular" part and the "triangular" part. The center of gravity (CG) of an object could also be its dividing point, and a *back* and *front* parts of the object with respect to the CG can thus be identified, which will not coincide exactly with the identification based on the shapes. If the object is symmetrical (i.e., no "triangular" vs "rectangular" region), then these are more likely to be referred to as "left" and "right" parts.

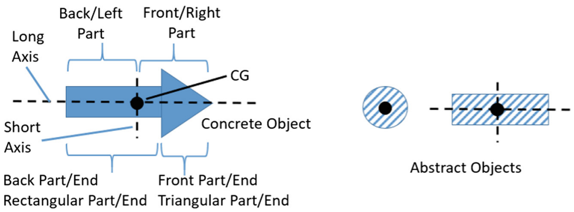

**Fig. 4.** An object and its parts. Definitions of Center of Gravity (CG), and long and short axes. Abstract objects are hatched and used in subsequent discussions.

Other than the CG, there are the "long axis" and "short axis" which could typically be identified for a longish object.

Predicate representations can be used to identify or refer to the various subparts of the object involved, such as ***Long-Axis***(Object1), ***Front-Part***(Object1), ***CG***(Object1), etc.

A concrete object such as the arrow-shaped object have sub-structures that are identifiable as its parts. In the following discussion, we will also use "abstract" objects, which could represent "any" objects, and these are shown as hatched shapes in Fig. 4 – the circular one represents something that is more or less symmetrical with no clear distinction of a *long* and *short axes*, and in the rectangular one, *long* and *short axes* can be discerned.

***Spatial Relationships***

Spatial relationships are very fundamental to the positioning and placement of objects and entities for the purpose of subsequent goal-oriented actions. Figure 5 shows some very fundamental spatial relationships.

Figure 5(a) shows the "above" relationship. In the spirit of cognitive linguistics as discussed in Sect. 2.3, the focus of the relationship is the Trajector, TR, and the reference, the Landmark, LM. In a sentence like, "A is *above* B," A is the TR and B is the LM. And as shown in Fig. 5(a), when a relationship such as "TR is above LM" is specified, there could be a range of locations in which TR can be positioned with respect to LM, and this is shown as two arrows showing the tolerable relative positions of the CG in which the *above* relation still holds. However, if more specific relationships are specified, such as "TR is 5 cm above LM," then the location range is more constrained.

These tolerable relative positions may be derived from observational statistics of the relative positions of the real-world instances of objects in which the relationship of *above* holds. These probabilities can then be used to determine what is a most "typical" configuration for the *above* spatial relationship.

Specifications such as this in Fig. 5(a) and in the rest of the figures can be used for **recognition** as well as **generation**. For recognition, the specification is mapped onto two real world objects, and if they satisfy this specification, they are in the above spatial relation.

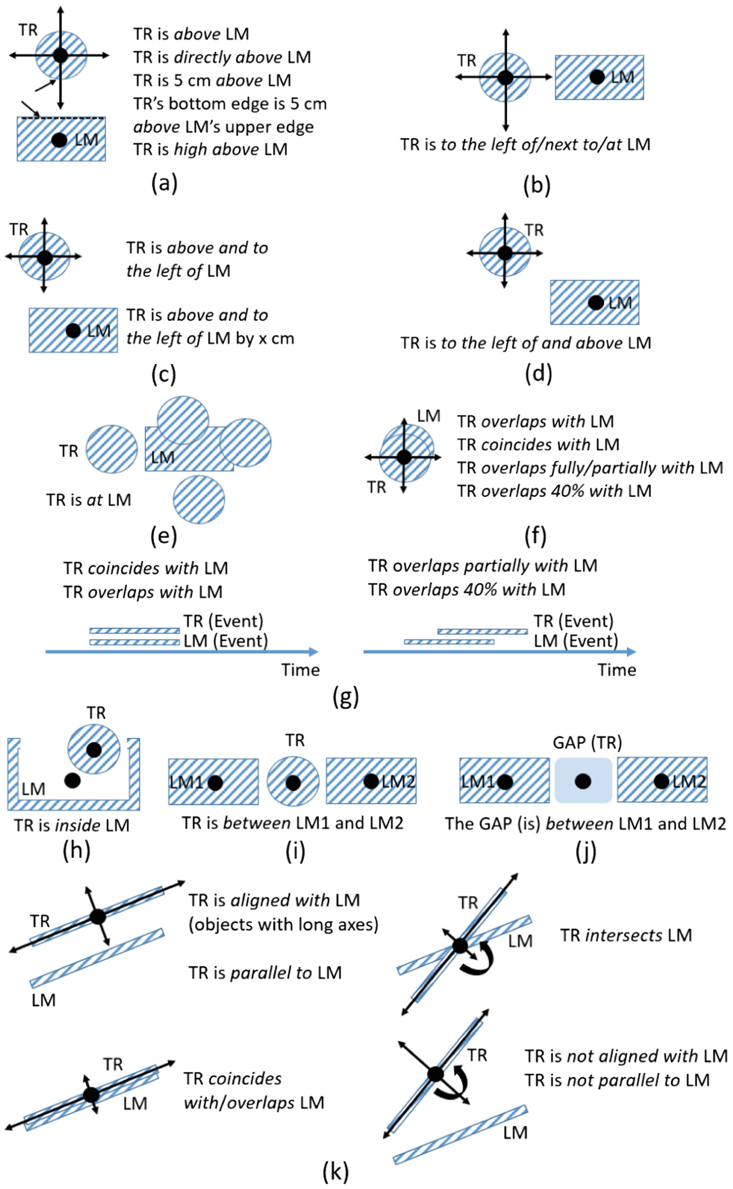

**Fig. 5.** Spatial relations. (a) *above*; (b) *to the left* or *right/next to/at*; (c) *above and to the left of*; (d) *to the left of and above*; (e) *at*; (f) *overlap/coincide with*, fully or partially; (g) temporal *overlap/coincide with*, fully or partially; (h) *inside*; (i) *between*; (j) a GAP as an object, *between*; (k) entities with longish spatial extent. The horizontal and vertical double arrows indicate the ranges of positions of the corresponding CGs.

For generation, the specification would direct the placement of the objects involved accordingly, using the concepts as well as probabilities learned earlier as a guide to produce the "typical" situation, taking consideration of other constraints (e.g., there may be other blockages so that the next most probable placement is selected).

Figure 5(b) shows the "to the left of" (and similarly "to the right of") relationships. If the objects are very close together, the description could be "next to" or "at."

Figure 5(c) shows a combination of *above* and *left of.* The situation is more of an *above* than a *left of*, so the word order is *above and to the left of.* If the situation is more *left of* than *above*, than it may be more likely to be described as *left of and above* such as shown in Fig. 5(d). Note that in both Figs. 5(c) and (d), the range of allowable positions of the TR is smaller than those in Fig. 5(a) and (b).

Figure 5(e) shows that other than the very nearby kind of *left of* or *right of*, even if there is some overlap between the TR and LM, the relationship could be "the TR is *at* the LM." Figure 5(f) shows the "overlap" situation. Sometime, "coincide" may be used to describe the same situation.

Figure 5(g) shows the concept of *overlap/coincide* applied to a temporal situation in which one *event* is a TR and another an LM. One can also say "Event1 (TR) is *next to* Event2 (LM)" but there is no corresponding situation of "Event1 is above Event 2."

Figure 5(h) shows a situation in which the LM is a container-like object and the TR is likely to be described as "inside" the LM rather than just *overlap* with the LM. All the other relationships of *above*, *to the left*, *to the right*, *next to*, etc. can also be characterized as "outside," if LM is container-like.

Figure 5(i) shows the "between" relationship. Figure 5(j) shows how a "gap" could be characterized as an "object" and the relationship of *between* could be applied here.

Figure 5(k) shows entities with spatial extents in which the relative orientation between the entities is important. In the case of entities with a longish spatial extent (i.e., a long axis can be defined), they can be *in parallel* to each other or *aligned* with each other, or *not in parallel* or *not aligned* with each other. The entities can also *intersect* each other or *coincide/overlap with* each other as shown in the same figure. Even though *intersect* is a little like *overlap*, for longish objects in which the "overlapping" area is small, *intersect* is used to describe the relationship instead.

### ***Sentence to Action Predicates Transformation***

Sentences are "surface" structures and different languages may represent the same "deep" structure (i.e., "meaning") using different surface structures (e.g., in some languages such as English, the word order is SUBJECT—VERB—OBJECT, while in some other languages, e.g., Japanese, it could be SUBJECT—OBJECT—VERB). Even within the same language, a passive voice and an active voice surface structure map onto the same deep structure (e.g., Mary *kicks* John has the same "meaning" as John is *kicked* by Mary). Whatever the surface structure, we should have transformation rules that map them into the same deep structure if they are indeed the same in meaning. In the following, we show an example of a Spatial Predicate and an example of an Action Predicate created from the surface structure sentences.

TR is *above* LM
→ **Above**(TR, LM) – **Spatial** Predicate

Mary *kicks* John **or** John is kicked by Mary
→ **Kick**(John, Mary) – **Action** Predicate

Next, we show an example of the use of the concept of **Until**. (Fig. 3)

John *moves* his hand *until* it is next to the Wall
→ **Move**(John, *Hand*(John), **Until**(*Next-To*(*Hand*(John), Wall)

Usually what follows **Until** is a condition to be met to terminate the earlier action. And usually, the concept of "positioning" is used to describe a similar situation more succinctly as follows:

John *positions* himself at the door
→ **Move**(John, John, **Until**(*At*(John, Door)

Note that the **Move** Action Predicate could be a simple movement or the more complex sequence of steps generated through a problem solving process – i.e., the **Until** predicate specifies a Goal State, and a problem solving process is called to generate the sequence of move actions to reach that state from the current state.

***Predicate to Referent Conversion: Representation and Action***

Having derived the predicate descriptions from sentences as described above, the next step of processing is to convert the predicates to their referents. One kind of predicate we discussed above is a predicate that specifies a subpart of an object. Figure 6 depicts an example of referencing the *Long-Axis* of an object.

Figure 6 shows that in the process of referencing the *Long-Axis* of an object, the equation of the *Long-Axis* is returned (some other forms of representation, such as an analogical representation, for the long-axis may be used instead of an equation). Similarly, something like **Top-Surface**(ObjectA) will return the equation or some other representational form for the top surface. If the predicate is **Front-Part**(ObjectA), then a volumetric representation of that front part is returned.

Figure 7 shows how a spatial predicate is converted into a referent. Suppose the spatial predicate *Above* is used to describe the spatial relationship between two specific objects, ObjectA and ObjectB. The process begins with the recall of the grounded representation of *Above*, shown on the left side of the figure (see Fig. 5(a)). Then the specific concrete instances ObjectA and ObjectB are bounded to the respective abstract objects in the grounded representation, and a *grounded and specific* concrete representation of *Above*(ObjectA, ObjectB) is output. This specific concrete representation of *Above*(ObjectA, ObjectB) contains a range of possible positions for ObjectA, relative to ObjectB.

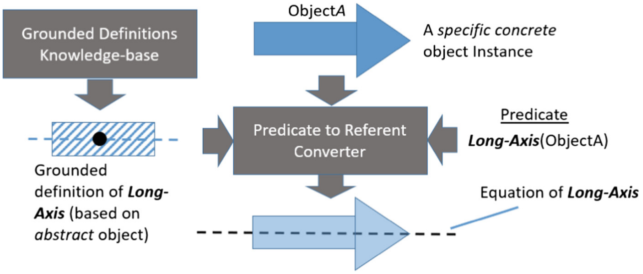
Grounded Definitions Knowledge-base
ObjectA
A specific concrete object Instance
Predicate to Referent Converter
Predicate
Long-Axis(ObjectA)
Grounded definition of Long-Axis (based on abstract object)
Equation of Long-Axis

**Fig. 6.** Predicate to Referent Converter: An **Aspect-of-Object** Predicate, ***Long-axis***, references the long axis of an ObjectA, and the Predicate to Reference Converter outputs the equation or other forms of representation of the long axis.

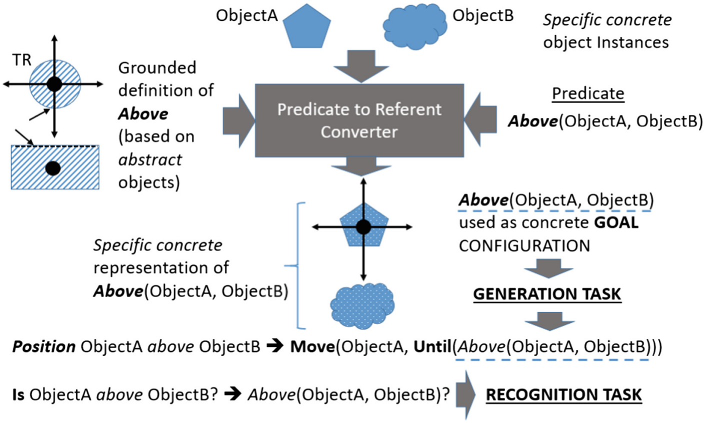
ObjectA
ObjectB
Specific concrete object Instances
TR
Grounded definition of Above (based on abstract objects)
Predicate to Referent Converter
Predicate
Above(ObjectA, ObjectB)
Above(ObjectA, ObjectB) used as concrete GOAL CONFIGURATION
Specific concrete representation of Above(ObjectA, ObjectB)
GENERATION TASK
Position ObjectA above ObjectB → Move(ObjectA, Until(Above(ObjectA, ObjectB)))
Is ObjectA above ObjectB? → Above(ObjectA, ObjectB)?
RECOGNITION TASK

**Fig. 7.** Predicate to Referent Converter: A **Spatial Predicate *above*** is converted into the corresponding referent. Note the distinction between a RECOGNITION TASK and a GENERATION TASK, and the corresponding predicate representations are shown.

The use of this specific representation can be for answering a query such as "Is ObjectA above ObjectB?", i.e., "*Above*(ObjectA, ObjectB)?" For this task, this concrete representation can be used to match to the current specific ObjectA and ObjectB in their corresponding specific locations to see if the *Above* relationship really holds. This is a RECOGNITION TASK.

This specific representation can also be used for a GENERATION TASK. I.e., suppose currently ObjectA and ObjectB are not in an *Above* relationship. The specific concrete representation of *Above*(ObjectA, ObjectB) is then used to specify a desired GOAL CONFIGURATION for a problem solving process to take actions to achieve that goal.

Because there is still a range of possible positions of ObjectA relative to ObjectB, before the problem solving process can use the goal, it may take the most typical relation positions (such as ObjectA is **directly** above ObjectB), or if the *Above* relationship has further specification arguments such as *Above*(ObjectA, ObjectB, **Directly**), *Above*(ObjectA, ObjectB, **5 cm**), etc., it will use those arguments to restrict the range of the possible positions of ObjectA relative to ObjectB or use those arguments as a basis for further questioning of the command giver (e.g., the human involved) for further specific instructions. The system may also have knowledge, for a given context, about the exact relative positions of ObjectA and ObjectB, given the *Above* relationship.

### Integration

Armed with the above basic constructs, we are now ready to describe how the system may process the complex instructions (2) from Sect. 3.1.

Figure 8 depicts a robotic hand with fingers and a screw driver with two subparts, the *Handle* and the *Shank*. The operation instructed by the first sentence "position your hand directly above the screw driver's handle" is shown. Through the Sentence to Predicate transformation process described above, this is translated into the following predicate representation:

**Move**(*Hand*(Robot), **Until**(*Above*(*Hand*(Robot), *Handle*(Screw-Driver), **Directly**)

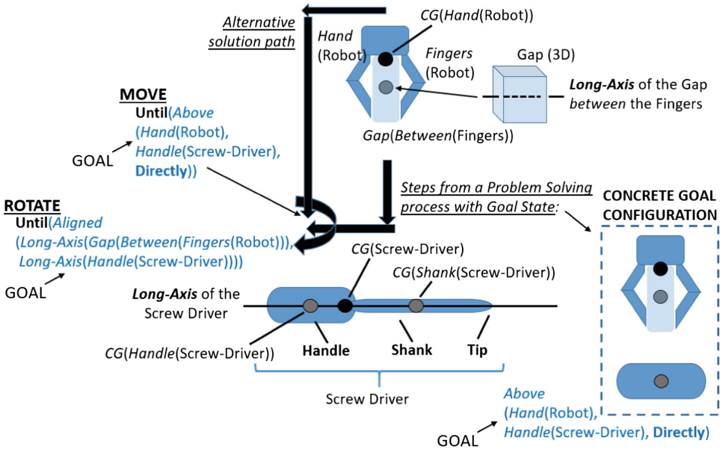
Alternative solution path
Hand (Robot)
CG(Hand(Robot))
Fingers (Robot)
Gap (3D)
Long-Axis of the Gap between the Fingers
Gap(Between(Fingers))
MOVE
Until(Above (Hand(Robot), Handle(Screw-Driver), Directly))
GOAL
Steps from a Problem Solving process with Goal State:
CONCRETE GOAL CONFIGURATION
ROTATE
Until(Aligned (Long-Axis(Gap(Between(Fingers(Robot))), Long-Axis(Handle(Screw-Driver))))
GOAL
CG(Screw-Driver)
CG(Shank(Screw-Driver))
Long-Axis of the Screw Driver
CG(Handle(Screw-Driver))
Handle
Shank
Tip
Screw Driver
Above (Hand(Robot), Handle(Screw-Driver), Directly)
GOAL

**Fig. 8.** A situation with a robot hand and a screw driver, and the robot is instructed to position its hand directly above the screw driver's handle.

To carry out this action, first the Predicate to Referent process discussed above and depicted in Fig. 7 is engaged to derive the CONCRETE GOAL CONFIGURATION of *Above*(*Hand*(Robot), *Handle*(Screw-Driver), **Directly)** as shown. Then, a problem solving process, employing either forward or backward chaining, is used to derive the action steps to bring the hand to the designated position. There could be more than one solution derived from the problem solving process.

Figure 9 illustrates the rest of the Sentence to Predicate conversion process and further Predicate to Referent conversion process will derive the corresponding actions.

Figure 8 also illustrates the goal of the **Rotate** action involved. **Rotate**…**Until** is also an action like the earlier **Move**…**Until** that will launch a problem solving process if needed. Note that in Fig. 9 a variant of the **Move** Action Predicate is introduced in which the direction of movement is specified in an argument.

A similar process can be used to process and understand the other complex sentence example (1) used in Sect. 3.1.

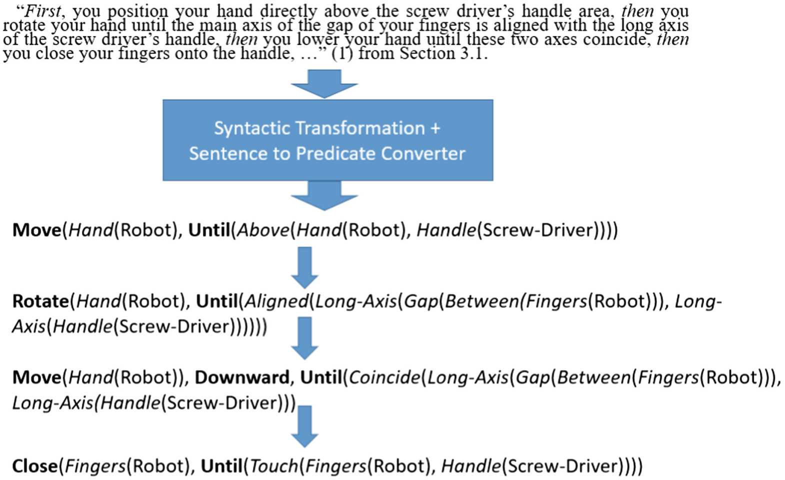
"*First*, you position your hand directly above the screw driver's handle area, *then* you rotate your hand until the main axis of the gap of your fingers is aligned with the long axis of the screw driver's handle, *then* you lower your hand until these two axes coincide, *then* you close your fingers onto the handle, …" (1) from Section 3.1.
Syntactic Transformation + Sentence to Predicate Converter
**Move**(*Hand*(Robot), **Until**(*Above*(*Hand*(Robot), *Handle*(Screw-Driver))))
**Rotate**(*Hand*(Robot), **Until**(*Aligned*(*Long-Axis*(*Gap*(*Between(Fingers*(Robot))), *Long-Axis*(*Handle*(Screw-Driver))))))
**Move**(*Hand*(Robot)), **Downward**, **Until**(*Coincide*(*Long-Axis*(*Gap*(*Between*(*Fingers*(Robot))), *Long-Axis(Handle*(Screw-Driver)))
**Close**(*Fingers*(Robot), **Until**(*Touch*(*Fingers*(Robot), *Handle*(Screw-Driver))))

**Fig. 9.** The complete Predicate representation of the complex sentence under consideration.

## 4 Conclusion and Summary

This paper analyzed the connection between language and robotics and successfully demonstrated how complex natural language instructions can be processed by a robotic system that will then carry out the actions accordingly. Two goals have been achieved at once for two seemingly disparate domains – it elucidates the meaning of meaning for the domain of linguistics and NLU, and it also elucidates the representational and computational processes for a robot to convert natural language instructions to actions.

This paper first reviewed and analyzed the previous works in linguistics, AI NLP and NLU, and robotics that point to a direction of how NLU can be applied to robotics. The paper then elucidates the steps of converting the temporal order embedded in certain natural language instructions to an explicit temporal representation for tasks to be carried out accordingly. Following that, a Sentence to Predicate conversion process is elucidated, followed by a Predicate to Referent process that grounds the meaning of the sentence in real-world constructs of spatial relations, spatial arrangements, and action sequences. This enables the robot to carry out the instructions accordingly.

Further work includes bringing in more of the cognitive linguistic constructs, such as those illustrated in Fig. 2, to represent the meaning of even more complex sentences, finessing the representational constructs for spatial relationship and spatial arrangements so that they can participate in the recognition and generation processes as discussed, and upgrade the 3 major blocks of processing, the Syntactic Transformation Rules, the Sentence to Predicate Converter, and the Predicate to Referent Converter, to handle more general and complex sentences. This will bring about the development of truly intelligent robots which can perform human-like understanding in the future.

## References

1. Taniguchi, A., Taniguchi, T., Cangelosi, A.: Cross-situational learning with Bayesian generative models for multimodal category and word learning in robots. Front. Neurorobotics **11**, 66 (2017). https://doi.org/10.3389/fnbot.2017.00066
2. Matuszek, C., Herbst, E., Zettlemoyer, L., Fox, D.: Learning to parse natural language commands to a robot control system. In: Desai, J., Dudek, G., Khatib, O., Kumar, V. (eds.) Experimental Robotics, pp. 403–415. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-319-00065-7_28
3. Khayrallah, H., Trott, S., Feldman, J.: Natural language for human robot interaction. In: Proceedings of the Workshop on Human-Robot Teaming at the 10th ACM/IEEE International Conference on Human-Robot Interaction (2015)
4. Ferrucci, D., et al.: Building Watson: an overview of the DeepQA project. AI Mag. **31**(3), 59–79 (2010)
5. Ganegedara, T.: Natural Language Processing with TensorFlow: Teach Language to Machines Using Python’s Deep Learning Library. Packt Publishing, Birmingham (2018)
6. Manning, C.D., Schutze, H.: Foundations of Statistical Natural Language Processing. MIT Press, Cambridge (1999)
7. Wang, Z., Chong, C.S., Lan, L., Yang, Y., Ho, S.-B., Tong, J.C.: Fine-grained sentiment analysis of social media with emotion sensing. In: IEEE Future Technologies Conference 2016 (FTC 2016), San Francisco, United States, 6–7 December 2016
8. Winograd, T.: A procedural model of language understanding. In: Schank, R., Colby, K.M. (eds.) Computer Models of Thought and Language. W. H. Freeman & Company, San Francisco (1973)
9. Ho, S.-B., Wang, Z.: On true language understanding. In: 5th International Conference on AI and Security, New York, 26–28 July 2019
10. Alomari, M., Duckworth, P., Hogg, D.C., Cohn, A.G.: Natural language acquisition and grounding for embodied robotic systems. In: Proceedings of the 31st AAAI Conference on Artificial Intelligence (2017)
11. Schank, R., Abelson, R.: Scripts, Plans, Goals, and Understanding. Lawrence Erlbaum Associates, Hillsdale (1977)
12. Pei, M., Jia, Y., Zhu, S.-C.: Parsing video events with goal inference and intent prediction. In: International Conference on Computer Vision. IEEE, New Jersey (2011)
13. Si, Z., Pei, M., Yao, B., Zhu, S.-C.: Unsupervised learning of AND-OR grammar and semantics from video. In: International Conference on Computer Vision. IEEE, New Jersey (2011)
14. Cruse, A.: Meaning in Language. Oxford University Press, Oxford (2011)
15. Langacker, R.W.: Foundation of Cognitive Grammar, vols. I and II. Stanford University Press, Stanford (1987)