What to do and how to do it:
Translating Natural Language Directives into Temporal and Dynamic Logic Representation for Goal Management and Action Execution

Juraj Dzifcak and Matthias Scheutz and Chitta Baral and Paul Schermerhorn

Abstract— **Robots that can be given instructions in spoken language need to be able to parse a natural language utterance quickly, determine its meaning, generate a goal representation from it, check whether the new goal conflicts with existing goals, and if acceptable, produce an action sequence to achieve the new goal (ideally being sensitive to the existing goals).**

In this paper, we describe an integrated robotic architecture that can achieve the above steps by translating natural language instructions incrementally and simultaneously into formal logical goal description and action languages, which can be used both to reason about the achievability of a goal as well as to generate new action scripts to pursue the goal. We demonstrate the implementation of our approach on a robot taking spoken natural language instructions in an office environment.

I. INTRODUCTION

Social and service robots have become quite advanced in their overall mechanical design and behavioral functionality, allowing them to perform quite sophisticated tasks (from delivering mail in office environments, to doing certain household chores, to allowing for simple play-like interactions not unlike those people have with their pets). Yet, natural language interactions with those robots are still in the very infancy, for several good reasons. Speech recognition has been a recurrent show stopper due the noise levels in natural environments where social and service robots are employed. Moreover, parsing, semantic interpretation, and dialogue management are typically only performed for a very limited set of natural language primitives, and thus allow for only a tiny set of natural language instructions that user could give to robots that would also be understood. This lack of sufficient natural language capabilities significantly diminishes the utility of many social and service robots and provides a main hurdle for deploying robots in open unconstrained natural environments (like office spaces and living rooms).

In this paper, we propose a novel way for processing natural language on robots that tightly integrates natural language (NL) with the robot's goal management and action execution systems. Specifically, we will develop an incremental NL parser that takes lexical items (from English) with syntactic annotations from a *combinatorial categorial grammar* and semantic annotations from *temporal and dynamic logics* extended by λ-expressions and maps them onto λ-free temporal and dynamic logic expressions that represent the goals and actions specified in the natural language directive, respectively. This two-pronged approach of extracting goals and action sequences *incrementally at the same time* has many advantages: (1) it allows for quick responses for the robot when it does not understand an expression (e.g., [6]), (2) it enables the robot to check the consistency of the new goals from the directives with existing goals, (3) it gives the robot an (at least partial) action sequence to achieve the goals, which can be further refined via planners, and (4) it allows the robot to detect and react to important syntactic, semantic, and pragmatic ambiguities in the directive immediately.

The rest of the paper proceeds as follows. We start with a very brief background of the employed temporal and dynamic logics, and give a few examples on how goals and action sequences can be specified based on NL directives. We also review some existing work on the processing of NL directives on robots. We then introduce the different steps in our approach of the parallel generation of temporal and dynamic logic formulas based on NL directives and describe the implementation in the context of our integrated robotic DIARC architecture. Finally, we show results from qualitative evaluations we performed on the robot in an office domain, discussing the properties and limitations of the current approach and concluding with a summary of our work and possible future directions.

II. BACKGROUND

The general idea of our proposed system is that the meaning of verbs in directives, instructions or commands can be interpreted as having two intended meanings: (1) that of specifying post-conditions or *goal states* and (2) that of specifying means of achieving them or *action sequences*. For example, the instruction "Go to the breakroom" specifies both the goal of *being at the breakroom at some future point in time* and the action of *going to the breakroom*. Consequently, we need to find both goal and action representations in a formal language to capture both meanings. Since goal specifications make intrinsic references to times, time intervals, events, etc., as well as temporal properties of behaviors, we need a logic that can capture temporal aspects. And since action specifications make reference to actions, sequences of actions, conditional actions, repeated actions, etc., we need an action logic to capture the dynamic aspects of actions and action execution. While there have been proposals for merging temporal and action logics that could express both aspects within the same formal framework

Juraj Dzifcak, ASU, jurajd@gmail.com
Matthias Scheutz, Indiana University, mscheutz@indiana.edu
Chitta Baral, ASU, chitta@gmail.com
Paul Schermerhorn, Indiana University, pscherme@indiana.edu

(e.g., TAL [7]), there are good reasons to separate the two aspects.[1]

We will thus first review the two formal languages/logics that we chose to specify goals and action sequences and then give examples of how natural language directives can be expressed in both formalisms.

A. Branching Temporal Logic CTL*

Branching temporal logics such as CTL* [8] were developed to be able to specify goals that cannot be specified using linear temporal logics like LTL.[2] The need for branching the time operators arises when we have to specify conditions outside the agent's path or plan. For example, when a robot is moving from position A to position B, we might require it to be always within a certain distance of a charging station. This goal cannot be expressed in LTL, which only has state formulas that are properties of states. Let $\langle p \rangle$ denote an atomic proposition, $\langle sf \rangle$ denote state formulas, and $\langle pf \rangle$ denote path formulas.

$$sf ::= p \mid sf \wedge sf \mid sf \vee sf \mid \neg sf \mid \mathsf{E}\ pf \mid \mathsf{A}\ pf$$
$$pf ::= sf \mid pf\ \mathsf{U}\ pf \mid \neg pf \mid pf \wedge pf \mid pf \vee pf \mid \bigcirc pf \mid \Diamond pf \mid \Box pf$$

The symbols A and E are the branching time operators meaning "for all paths" and "there exists a path" respectively. The branching structure is specified by a transition relation R between states of the world. Intuitively, $R(s_1, s_2)$ means that the state of the world can change from s_1 to s_2 in one step. Given a transition relation R and a state s, a path in R starting from s is a sequence of states $s_0, s_1, \ldots$ such that $s_0 = s$, and $R(s_i, s_{i+1})$ is true.

When planning in an environment where the robot is the only one that can make changes to the world, $R(s_1, s_2)$ is true if there exists an agent's action a such that $s_2 = \Phi(s_1, a)$. If there are external agents other than the robot then $R(s_1, s_2)$ is true if there exists an action (by some agent) a such that $s_2 = \Phi(s_1, a)$. Finally, we say a sequence of actions $a_1, \ldots, a_n$ is a plan with respect to the initial state s_0 and a goal G if $(s_0, R, \sigma) \models G$, where σ is the trajectory corresponding to s_0 and $a_1, \ldots, a_n$. For details on the formal semantics of CTL*, see [4].

B. Dynamic logic

Actions and action sequences that the robot can execute are specified in the form of *scripts* or *action programs*. The specification is based on a simplified subset of first-order dynamic logic (FDL) (e.g., [11] without the "?" operator that turns arbitrarily complex formulas into programs that can check their truth).[3] Rather, checking the truth of a formula must be achieved via special "primitive truth checking actions" that are defined for some predicates (e.g., there are special primitive actions that can check whether an obstacle is in front of the robot and thus explicitly test the truth of the expression "obstacle(infront)"). For all other formulas (that do not have corresponding primitive truth checking actions), checking the truth has to be accomplished via explicit *truth checking programs* (expressed in dynamic logic). For example, checking whether $\phi \vee \psi$ is true translates into the program "`if` $\neg\phi$ `then` ψ `else true`".[4] Programs are defined in the standard way (as regular expressions) based on a set of primitive actions Π that can be combined to form complex programs using the standard operations for *sequence* $(\alpha; \beta)$, *choice* $(\alpha \cup \beta)$, and *iteration* $(\alpha*)$. Conditionals (`if` ϕ `then` α `else` β) and conditional loops (`while` ϕ `do` α) are also defined in the usual way, hence we will not give a detailed description of FDL and its semantics here, but refer to the treatment in [11].[5] We have implemented an *action interpreter* which will take a program specified in the above restricted FDL and execute it (e.g.,[6]).

C. Examples of NL translations into goal descriptions and action scripts

We can now illustrate how a simple natural language instruction like "go to the breakroom and report the location of the blue box" in the context of an office domain can be expressed in our goal and action interpreter languages and how multiple possible translations can reveal interesting ambiguities in the goal specification based on the meaning of "and".

The first translation assumes that the intended meaning of "and" is that of a temporal sequence and thus that the robot is supposed to report the location of a blue box that located *within* the breakroom. This can be represented in CTL* as $\Diamond(at(breakroom) \wedge \Diamond reported(location, blue\_box))$ and the corresponding action script (i.e., the robot program) to accomplish these goals can be translated as $go\_to(breakroom); report(location, blue\_box)$.[6]

[1]For one, because the computational complexity of TAL expressions is intractable. Another reason is that building fast and incremental independent translations allows for early starts of actions (based on successfully parsed action primitives, e.g., see [6]), quick consistency checks of goals, dialogue-based disambiguation of instructions, partial understanding of sentence fragments (where either only goal or only actions are understood), and many others. For space reasons, we have to defer developing more fully the rationale for using separate logics.

[2]The role of LTL in specifying planning goals has been well studied and examples of that can be found in [1], [14], [3].

[3]The rationale for this restriction is that checking the truth of (complex) predicates is typically not an atomic operation, but takes time and amounts to executing actions in the virtual machine of the robot architecture (e.g., checking whether a goal has been reached will require a look-up operation on the goal stack, or checking whether an obstacle can be seen in front of the robot will require checking distance and/or vision sensors). As such, complex formulas might require a sequence of operations in the virtual machine architecture that may or may not be executed in parallel, and could potentially fail at different points. Hence, it seems reasonable in the robotic context to require that the processes of checking the truth of formulas be explicitly expressed in programs.

[4]Note that the negation here is also treated as an action that negates the exit status of an action.

[5]We are also using a variant of PDL with parallel execution $\alpha||\beta$ of two actions α and β which has been demonstrated to be finitely axiomatizable and decidable.

[6]Note that we assume here that the *report*-action takes two arguments, the first being a property of an object, the second being an object type. This is to avoid complications with quantifiers that arise from using determiners like "a", "the", "one", "any", "some", "all", etc. which all have different meanings and require different translations.

Another, quite different, goal description is obtained if "and" is construed "propositionally" (instead of specifying a sequence): $\Diamond at(breakroom) \wedge \Diamond reported(location, blue\_box)$ with the corresponding action script being $go\_to(breakroom)||report(location, blue\_box)$. In this case, the robot could report a blue box that it might see on the way to breakroom instead of a box located within the breakroom. If one wants the robot to report only a blue box within the room, then one needs to add an explicit statement that no report should be made before the robot is in the breakroom, which can be translated as $\Diamond(at(breakroom) \wedge \Diamond reported(location, blue\_box) \wedge \neg reported(location, blue\_box) \mathsf{U} at(breakroom))$. Note that while this translation will require that the report be made from within the breakroom, it still does not prevent the robot from sensing a blue box outside the room and only reporting its location once it is inside the room (to account for this possibility, additional predicates regarding the perceptual functions and when they are executed need to be added to the goal specification and to the program).

This simple example already demonstrates several important challenges with natural language directives, in particular, that they frequently involve default assumptions about how to interpret logical and temporal connectives, that they can be syntactically and semantically ambiguous, but that pragmatic constraints might hint at the intended meaning, and that logical translations might be one way for robots to address and deal with these ambiguities if the intended interpretation cannot determined (e.g., the robot could ask for clarification, providing the different interpretations it found, rather than just executing one of them).

D. Related Work

There are several examples of robots that can be given instructions or goals in natural language (e.g., a semi-autonomous wheelchair that responds to coarse route descriptions (e.g. "turn left," "follow corridor") [13]; or a robot that can be guided through an environment using goal-based commands (e.g. "go to the left of the ball") or direction-based commands ("turn right"); or the system developed by [9], which can use violations of preconditions of actions or impossible actions to eliminate incorrect interpretations and disambiguate otherwise ambiguous expressions; or the system proposed in [15] which can learn simple action scripts through natural language instruction as long as the instruction follows a clearly defined scheme or template). Yet, aside from a large body of research in natural language semantics on translating English into formal logics (e.g., first-order logic [5] or more recently ASP [2]), we are not aware of any examples of such NL systems on robots where natural language expressions are systematically translated into formal goal and action representations (even though some systems share the incremental processing properties of the proposed system, e.g., [12]).

In none of the above systems are natural language sentences mapped directly onto explicit goal representations that can be used both for checking the achievability of goals and as a guide for planning and action execution. Moreover, none of the above systems generates novel action scripts from NL expressions in a formal logic that can be directly used for verification of achievability (of the action sequence), for planning (to fill in missing actions due to the high level description), and for execution.

III. Incremental Natural Language Translation into Goal Languages and Action Scripts

We start with defining categories of all lexical items in a combinatorial categorial grammar ([18], [10]) and also introduce λ-expressions into CTL* and FDL to represent the goal and action meanings of lexical expressions.[7] At any given point in the parsing process (i.e., after the consumption of n lexical items), the parser will retrieve the grammatical category and associated λ-expressions for a new lexical item and determine the way its λ-expressions should be combined with those of the sentence fragment so far based on the order dictated by the CCG grammar (cp. to [2]).

A. Combinatorial Categorial Grammar (CCG)

Word or phrase	Categories
and, or, but, until, before	$(S/S)\backslash S$
while, always	$(S/S[c])\backslash S[c]$, $(S/S[c])/S[c]$
when	$(S/S[c])\backslash S[c]$, $(S/S[c])/S[c]$
within, during	$(S\backslash(S/NP)/S)\backslash S$, $(S/S)/S$
do not	$S/(S/NP)$, $(S/N)/(S/NP)$
go to, pass by, reach	(S/NP), $(S[c]/NP)$
stay, detect, clean, check	(S/NP), $(S[c]/NP)$
speak, wait	S, $S[v]$
is	$(S/NP)\backslash NP$
turn on, get, open, close	S/NP, $(S[c]/NP)$
report	$(S/PP[of])/NP$
a, the, one	NP/N
all, some	$NP/N[p]$
in	$NP/N[loc]$
of	$PP[of]/NP$
immediately, eventually, then	$(S/N)/(S/NP)$, $S/S[v]$
keep, maintain, stay	$(S/(S/NP))/N$, S/NP
you	$PP/(S/NP)$
ever	$(S\backslash(S/(S/NP)))/S$
Room 1...n_1	NP, N, $N[loc]$
Door 1...n_2	NP, N, $N[loc]$
Corridor 1...n_3	NP, N, $N[loc]$
hit, foyer, light, break, blue box	NP, N
blue boxes	$NP[p]$, $N[p]$
door, corridor, location	NP, N
recharge station, breakroom	NP, N, $N[loc]$
on, occupied, detected	NP, N

TABLE I
Combinatorial Categorial Grammar

Following [10], a *combinatorial categorial grammar* (CCG) can is characterized by (1) a set of *basic categories*, (2) a set of *derived categories*, each constructed from the basic categories, and (3) some syntactical (combinatorial) rules describing the concatenation and determining the category of the result of the concatenation.[8]

Table I shows a subset of the implemented lexical items from the office domain (i.e., words or phrases) and their

[7] Note that there is a long tradition in formal linguistics of using λ-calculus for translating English sentences into first order logic formulas [5].

[8] There are various combinatorial rules used in CCGs for natural language, such as (forward/backward/forward-crossing/backward-crossing) function application, (forward/backward/forward-crossing/backward-crossing) substitution and others (See, e.g., [18]). For the purpose of this presentation, we only assume forward and backward application rules.

Word or Phrase	λ-TL-expression	λ-FDL-expression
and	$\lambda x\lambda y.\Diamond(x \wedge \Diamond y)$ $\lambda x\lambda y.\Diamond x \wedge \Diamond y$	$\lambda x\lambda y.x; y$ $\lambda x\lambda y.x \mid\mid y$
but	$\lambda x\lambda y.\Diamond x \wedge \Diamond y$	$\lambda x\lambda y.x \wedge y$
or	$\lambda x\lambda y.\Diamond x \vee \Diamond y$	$\lambda x\lambda y.x \cup y$
until	$\lambda x\lambda y.x \mathsf{U} y$	$\lambda x\lambda y.x*; y$
before	$\lambda x\lambda y.\Diamond(y \wedge \Diamond x)$	$\lambda x\lambda y.x; y$
when	$\lambda x\lambda y.\Box(x \Rightarrow y)$	$\lambda x\lambda y.x \mid\mid y$
while	$\lambda x\lambda y.x \wedge y$	$\lambda x\lambda y.x \mid\mid y$
eventually, then	$\lambda x.\Diamond x$	$\lambda x.x$
immediately	$\lambda x.\bigcirc x$	$\lambda x.x$
reach, go to, pass by	$\lambda x.at(x)$, $\lambda x.\Diamond at(x)$	$\lambda x.go\_to(x)$
get	$\lambda x.\Diamond get(x)$	$\lambda x.get(x)$
stay	$\lambda x.\Box at(x)$, $\lambda x.\Box x$	$\lambda x.stay - at(x)$
report	$\lambda x\lambda y.report(x, y)$	$\lambda x\lambda y.report(x, y)$
maintain, ever, always	$\lambda x.\Box x$	$\lambda x.x$
keep	$\lambda x.\Box x$	$\lambda x.x$
within	$\lambda x \mathsf{E} x$	$\lambda x.x$
during	$\lambda x \mathsf{E} x$	$\lambda x.x$
do not	$\lambda x.\neg x$	$\lambda x.\neg x$
turn on	$\lambda x.on(x)$	$\lambda x.turn - on(x)$
detect	$\lambda x.detect(x)$	$\lambda x.detect(x)$
check	$\lambda x.check(x)$	$\lambda x.check(x)$
clean	$\lambda x.clean(x)$	$\lambda x.clean(x)$
open	$\lambda x.open(x)$	$\lambda x.open(x)$
speak	$\lambda x.x@speak$	$\lambda x.x@speak$
a, the, you, one	$\lambda x.x$	$\lambda x.x$
all, some, in, of, is	$\lambda x.x$	$\lambda x.x$
Room A	$\lambda x.x@roomA$	$\lambda x.x@roomA$
Corridor B	$\lambda x.x@corridorB$	$\lambda x.x@corridorB$
Door C	$\lambda x.x@doorC$	$\lambda x.x@doorC$
light	$\lambda x.x@light$	$\lambda x.x@light$
on	$\lambda x.on(x)$	$\lambda x.on(x)$
blue box	$\lambda x.x@blue\_box$	$\lambda x.x@blue\_box$
recharge station	$\lambda x.x@recharge - station$	$\lambda x.x@recharge - station$
breakroom	$\lambda x.x@breakroom$	$\lambda x.x@breakroom$

TABLE II

SOME OF THE λ-EXPRESSIONS USED TO OBTAIN THE LOGICAL REPRESENTATIONS OF SENTENCES. WE ASSUME x AND y ARE ACTIONS, WITH ACTION-PREDICATE MAPPING AVAILABLE.

assigned categories in a standard NL CCG. Note that lexical items here do not include inflected forms (e.g., on verbs or plurals on nouns).

B. λ-calculus

In addition to grammatical categories, we also need to assign meanings to lexical items, which are $lambda$-$expressions$ in CTL* and FDL. A λ-expression is either a *variable* v, or an *abstraction* $(\lambda v.e)$ where v is a variable and e is a λ-expression; or an *application* e_1e_2 where e_1 and e_2 are two λ-expressions. Given a λ-expression e and variables x_1 and x_2, $\alpha(e, x_1, x_2) = e[x1 := x2]$, where $e[x1 := x2]$ denotes the substitution of x_1 by x_2 in e. Given a free variable x and λ-expressions e_1 and e_2, $(\lambda x.e_1)@e_2 = e1[x := e_2]$. A β-reduction can be viewed as a function application and will be denoted by the symbol @. For example, $\lambda x.go\_to(x)$ @ $breakroom$ results in $go\_to(breakroom)$.

Table II shows the $lambda$-expressions for CTL* and FDL associated with a subset of lexical items from the dictionary. A, B and C are variables corresponding to the indexes of the rooms, corridors and doors respectively. Please note we use the formula $\lambda x.x$ to denote that the word has no meaning for our logic(s). Also, we allow multiple λ-TL-expressions and action associations for lexical items with the same CCG category to capture lexical ambiguities (i.e., multiple meanings).

C. Parsing

We currently employ a simple heuristic-based parser that supports several combinatorial rules, most notably the forward and backward application and their generalized version. The parser incrementally builds goal and action descriptions for natural language directives by combining expressions based on those combinatorial rules using various heuristics. The first heuristic tries to assign categories in a such a way that for any possible complex category of a word, we initially try the ones for which all the basic categories (or at least the less complex ones) are present in the rest of the sentence. For example, if for a word 'w' we have the categories $S/(S/NP)$ and $(S/N)/(S/NP)$ to choose from and currently no word in the sentence has the category N assigned, we try the category $S/(S/NP)$ first.

Another heuristic is used for rule selection where less complex categories are combined first, if possible. This is because such combinations are more likely lead to a failure if there is one.

To get an understanding of how the parser obtains the final CTL* and FDL expression, we give an example of one of the derivations.

1) Go to the breakroom and report the location of the blue box. One of the possible derivations is shown in Table III. It shows the resulting formulas to be $\Diamond(at(breakroom) \wedge \Diamond report(location, blue\_box))$ and $go\_to(breakroom); report(location, blue\_box)$. Please note that there is an alternate derivation using a different formula for 'and', resulting in $\Diamond at(breakroom) \wedge \Diamond report(location, blue\_box)$ and $go\_to(breakroom) || report(location, blue\_box)$.

D. Implementation and Preliminary Evaluation

We used our DIARC [6] (based on the ADE robotic infrastructure [16]) for the implementation and evaluation of the proposed parser.[9] Specifically, the parser was integrated into our DIARC architecture [6] by replacing the previous incremental parser in the natural language subsystem of DIARC [6]. The architecture was run on a dual-core 2.4GHz Pentium Mobile Lenovo laptop under Linux kernel 2.6.24. The laptop was mounted on top of an ActivMedia Pioneer AT with a BumbleBee stereo fire-wire camera mounted above a Sick Laser. The laser was used for localization and obstacle avoidance, the camera for detecting colored objects with simple shapes (like boxes).

The test environment (see Fig. 1) was an office setting with a long hallway and several rooms on the right and the left of the hallway. The robot had an annotated map of the whole environment and was thus able to associate locations like "breakroom" with particular areas on the map. The robot received instructions via a wireless microphone

[9] DIARC is a distributed integrated affect reflection and cognition architecture especially developed for natural human-robot interaction that has been used with a variety of robots (from ActivMedia Pioneer and Peoplebots, to Segway robots, and various custom-made platforms). It integrates typical cognitive tasks (such as natural language understanding and complex action planning and sequencing) with lower level activities (such as multi-modal perceptual processing, feature detection and tracking, and navigation and behavior coordination) and has been used for several years in human subject experiments to study advanced human-robot interactions.

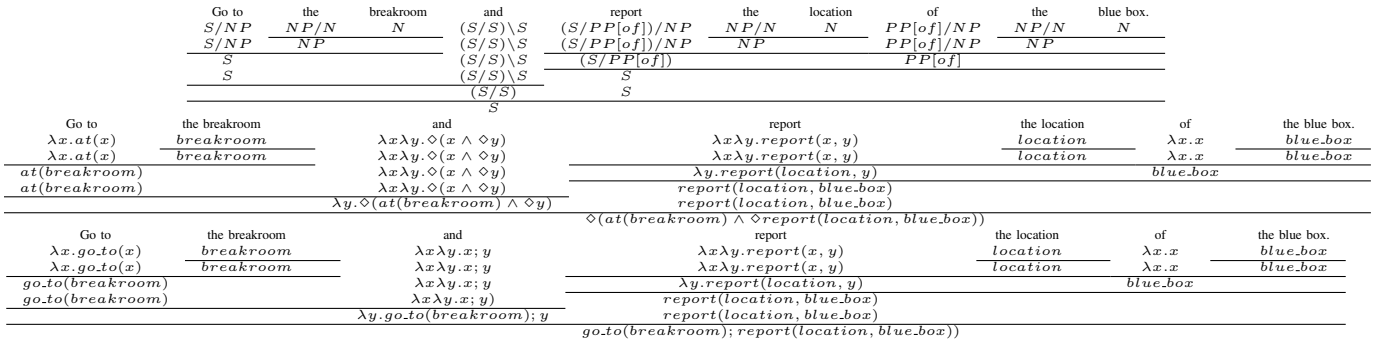

TABLE III

CCG AND λ-CALCULUS DERIVATION FOR "GO TO THE BREAKROOM AND REPORT THE LOCATION OF THE BLUE BOX."

which was connected to the sound card on the laptop onboard the robot.[10]

A human speaker instructed the robot in natural language to "go to the breakroom and report the location of the blue box". As soon as the utterance was finished, the robot had generated the goal representation $\Diamond(at(breakroom) \wedge \Diamond reported(location, blue\_box))$ and accepted the goal (as there was no other higher priority goal in conflict with it, see [6] for more details on the goal management subsystem). The robot then acknowledged the goal verbally ("OK, going to breakroom") and started to move towards the breakroom based on the action script $go\_to(breakroom); report(location, blue\_box)$. Once it arrived in the breakroom, it started a "look-for" action as part of the "report" action, found a blue box next to the printer (which was a landmark in its internal map of the office) and then generated a verbal report of the location "The blue box is by the printer" (based on the determined proximity of the target object to the closest landmark object, in this case the printer). We also tested the robot with various other similar instructions (e.g., the above example "Go to Room 7 and wait until the light is on") and in all cases the robot was immediately able to understand and carry out the directives.

IV. DISCUSSION AND RELATED WORK

The qualitative experimental evaluation demonstrated that our approach to generating goal and action representations from natural language expressions is viable (i.e., can be done in real-time on an actual robot in a natural human environment) and allows the robot to carry out directives that have temporal aspects and conditions (such as the ones described above) and can have some lexical and syntactic variation. For example, "move to the break-room and then wait" and "wait after you've reached the break-room" will result in the same interpretation of goals and programs (template-based robotic NL systems cannot handle these lexical and syntactic differences, e.g.,[15]). Yet, the current system is clearly only a start. For one, because natural language directives, aside from being ungrammatical or using words that are not in the robot's lexicon, can be incomplete and/or too abstract to allow for clear determinations of goals or actions. For example, our current system assumes that instructions are complete and grammatical (e.g., that there are no missing words, no wrong word substitutions, no ungrammatical syntactic constructs, etc.). Moreover, it assumes that all words are in the robot's lexicon and thus have a clearly specified grammatical category and semantic interpretation. Even under those assumptions, there are interesting problems connected to even simple instructions.

Take again the sentence "go to the breakroom and report the location of the blue box". Aside from the already mentioned ambiguity in the interpretation of "and" and the implicit assumption that the blue box is in the breakroom, the action script we obtain from the parser is actually incomplete in that it leaves out the "look-for" action which is required for the robot to determine the location of the blue box. In our experimental evaluation, we addressed this problem by making "look-for" a subgoal of the "report" action, but "report" could have solved this problem itself and determined online (via a planner or problem-solver) that it needs to perform the "look-for" action if we assume that $report(x, y)$ has a precondition $known(x, y')$ where y' in an instance of type y and that "look-for" has $\neg known(x, y')$ as precondition and $\neg known(x, y')$ as postcondition for perceivable properties x.

In addition to filling in implicit steps, there are challenges for determining the correct meaning (and thus the right parse) for lexically ambiguous words. Take again the conjunction "and" with its two interpretations (propositional vs temporally sequential). If the above instruction had been "Go to the breakroom and report the location of the blue box along the way", then it would have been clear that the blue box was not in the breakroom, and that both actions ("goto" and "report") had to be executed in parallel. Another example of how the intended meaning of "and" can depend on subsequent words (even based on the meaning of verbs) would be "go to the breakroom and remain undetected" (parallel execution) and "go to the breakroom and remain there" (sequential execution); in these cases the correct meaning cannot be determined until the final word.

A successful NL system that can take instructions from humans in (largely unconstrained) natural language will clearly

[10]For speech recognition we used CMU's SPHINX recognizer at http://cmusphinx.sourceforge.net/html/cmusphinx.php.

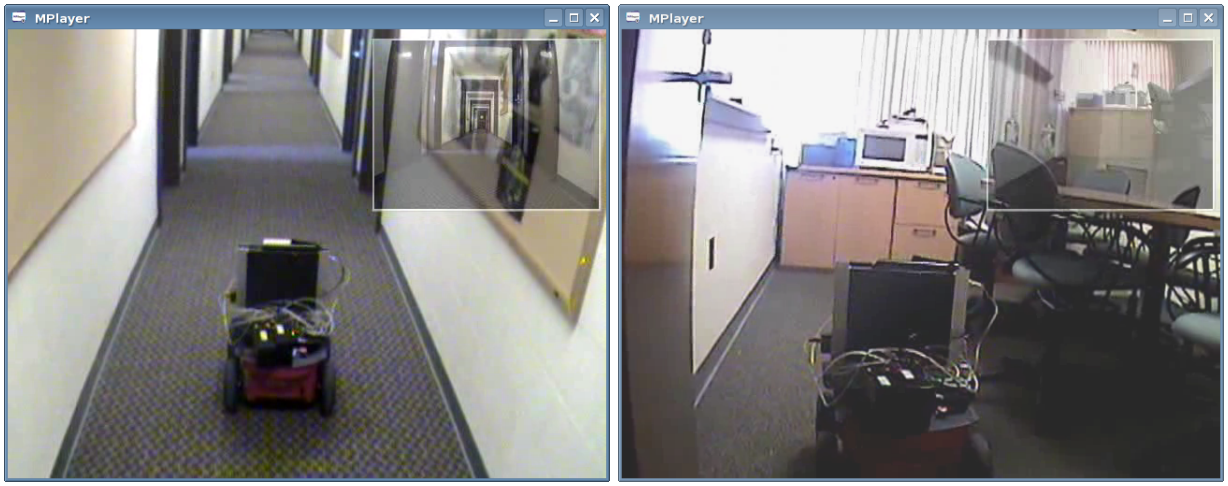

Fig. 1. The Pioneer AT robot used in the experiment when it received the instruction "Go to the breakroom and report the location of the blue box" in the hallway (left) and when it detected the blue box by the printer in the breakroom (right). The right upper corner of each screen shot shows the environment from the perspective of the robot's cameras.

have to address these and other challenges (e.g., including the parsing and generation of referential expressions in ways that do not lead to overspecification, see [17]).

V. CONCLUSION

In this paper we demonstrated a novel natural language translation scheme that allows robots to generate formal goal and action descriptions in temporal and dynamic logics from natural language directives. We have implemented a parser for the scheme and demonstrated on robot that it works very effectively in real-time for a small lexicon. We also discussed the advantages of such as system for planning, the detection of goal inconsistencies or ambiguities in the natural language specifications of the goals. The current system is clearly only a start and much more work on integrating annotations of lexical items using the employed logics is required to handle more complex instructions and cases of ambiguities. Future work will address the integration of the NL components with a planner than can determine and fill in missing steps in action scripts and with a natural language dialogue system that can be used to generate natural language questions to disambiguate or further specify insufficiently precise instructions. Moreover, a formal human-subject evaluation of the system is planned with different speakers under controlled and uncontrolled conditions.

VI. ACKNOWLEDGMENTS

This work was in part funded by ONR MURI grant #N00014-07-1-1049 to second and third author.

REFERENCES

[1] Fahiem Bacchus and Froduald Kabanza. Planning for temporally extended goals. *Annals of Math and AI*, 22:5–27, 1998.

[2] Chitta Baral, Juraj Dzifcak, and Tran Cao Son. Using ASP and lambda calculus to characterize NL sentences with normatives and exceptions. In *AAAI*, 2008.

[3] Chitta Baral, Vladik Kreinovich, and Raul Trejo. Computational complexity of planning with temporal goals. In *IJCAI-01*, pages 509–514, 2001.

[4] Chitta Baral and Jicheng Zhao. Goal specification in presence of non-deterministic actions. In *Proceedings of ECAI'04*, pages 273–277, 2004.

[5] Patrick Blackburn and Johan Bos. *Representation and Inference for Natural Language: A First Course in Computational Semantics*. Center for the Study of Language and Inf, 2005.

[6] Timothy Brick and Matthias Scheutz. Incremental natural language processing for HRI. In *Proceedings of the Second ACM IEEE International Conference on Human-Robot Interaction*, pages 263–270, Washington D.C., March 2007.

[7] Patrick Doherty, Joakim Gustafsson, Lars Karlsson, and Jonas Kvarnström. Temporal action logics (TAL): Language specification and tutorial. *Linköping Electronic Articles in Computer and Information Science*, 3(15), 1998.

[8] E. Allen Emerson. Temporal and modal logic. In J. van Leeuwen, editor, *Handbook of Theoretical Computer Science*, volume B, pages 997–1072, 1990.

[9] Will Fitzgerald and R. James Firby. The dynamic predictive memory architecture: Integrating language with task execution. In *Proceedings of the IEEE Symposia on Intelligence and Systems*, Washington, D.C, 1998.

[10] L.T.F. Gamut. *Logic, Language, and Meaning*. The University of Chicago Press, 1991.

[11] Robert Goldblatt. Parallel action: Concurrent dynamic logic with independent modalities. *Studia Logica*, 51(3/4):551–578, 1992.

[12] Geert-Jan M. Kruijff, Pierre Lison, Trevor Benjamin, Henrik Jacobsson, and Nick Hawes. Incremental, multi-level processing for comprehending situated dialogue in human-robot interaction. In *Language and Robots: Proceedings from the Symposium (LangRo'2007)IJCAI-01*, pages 509–514, 2007.

[13] R. Müller, T. Rofer, A. Landkenau, A. Musto, K. Stein, and A. Eisenkolb. Coarse qualitative description in robot navigation. In C. Freksa, W. Braner, C. Habel, and K. Wender, editors, *Spatial Cognition II*, pages 265–276. Spinger-Verlag, Berlin, 1998.

[14] Rajdeep Niyogi and Sudeshna Sarkar. Logical specification of goals. In *Proc. of 3rd international conference on Information Technology*, pages 77–82, 2000.

[15] P. E. Rybski, J. Stolarz, K. Yoon, and M. Veloso. Using dialog and human observations to dictate tasks to a learning robot assistant. *Journal of Intelligent Service Robots - To appear*, 2008.

[16] Matthias Scheutz. ADE - steps towards a distributed development and runtime environment for complex robotic agent architectures. *Applied Artificial Intelligence*, 20(4-5):275–304, 2006.

[17] Matthias Scheutz, Kathleen Eberhard, and Virgil Andronache. A parallel, distributed, realtime, robotic model for human reference resolution with visual constraints. *Connection Science*, 16(3):145–167, 2004.

[18] Mark Steedman. *The syntactic process*. MIT Press, 2000.