2014 IEEE International Conference on Robotics & Automation (ICRA)
Hong Kong Convention and Exhibition Center
May 31 - June 7, 2014. Hong Kong, China

Perceptive Feedback for Natural Language Control of Robotic Operations

Yunyi Jia, Ning Xi, Joyce Y. Chai, Yu Cheng, Rui Fang and Lanbo She

***Abstract*— A new planning and control scheme for natural language control of robotic operations using the perceptive feedback is presented. Different from the traditional open-loop natural language control, the scheme incorporates the high-level planning and low-level control of the robotic systems and makes the high-level planning become a closed-loop process such that it is able to handle some unexpected events in the robotics system and the environment. The experimental results on a natural language controlled mobile manipulator clearly demonstrate the advantages of the proposed method.**

I. INTRODUCTION

Recent years have seen a growing interest in technology on natural language control of robotic operations. The applications of such systems range from service and entertainment robots to industrial robots [1-3].

Natural language carries high-level discrete symbolic information and robot control is a low-level continuous process. Controlling robotic operations by natural language requires a mechanism to connect these two different representations. Despite recent progress, research on natural language control of robotic systems still faces many challenges. For example, most previous work has focused on grounding linguistic terms to the robot's perception and action [4-7] so that a natural language command can be translated to formal representations accessible by the robot (e.g., sensorimotor skills of the robot). However, an important issue which has not been well addressed is that, even given a perfect translation, unexpected events could happen that will prevent the execution of the natural language command. For example, suppose a human says "pick up the red bottle" and the robot understands perfectly what actions are involved (i.e., coming up with a task schedule given the current situation) and which bottle needs to be picked up. However, the situation may change after the human's command, for example, the robotic arm could be blocked or the object may be moved. The robot would fail in executing the command. Thus it is important to have a scheme that will allow the natural language controlled robotic system to intelligently handle unexpected events in both the robotic system and the environment.

This research work is partially supported under U.S. Army Research Office Contract No. W911NF-11-D-0001, and U.S. Army Research Office Grant No. W911NF-09-1-0321 and W911NF-10-1-0358, and National Science Foundation Award No. CNS-1320561 and IIS-1208390.

Yunyi Jia, Ning Xi and Yu Cheng are with the Department of Electrical and Computer Engineering, Michigan State University, East Lansing, MI 48824 USA (E-mails: jiayunyi@msu.edu, xin@egr.msu.edu, chengyu9@msu.edu).

Joyce Y. Chai, Rui Fang and Lanbo She are with the Department of Computer Science and Engineering, Michigan State University, East Lansing, MI 48824 USA (E-mails: jchai@cse.msu.edu, fangrui@cse.msu.edu, shelanbo@cse.msu.edu).

Researches on natural language control of robotic operations have been conducted for many years. In the human-robot interaction research, many studies have been focused on converting natural language commands to some types of logic representations and then mapping them to predefined primitives of robot actions [8-10]. Some studies have specifically looked into temporal logic [11] and spatial references [12] for natural language control. Natural language tasks are interpreted as these structured representations and many approaches have been proposed to generate the robot motions based on these representations and some abstracted plant models [13-16]. Recently, some studies have also used the probability-based methods to directly convert natural language commands to robot motions through off-line training or online learning of the robotic systems [17-20].

The approaches above have been mainly focused on generating the robot motions to achieve the tasks described by natural language. The high-level planning including the task scheduling and action planning and the low-level control of the robotic system are implemented as separate and sequential processes. This formalism usually needs to assume that the control system can absolutely accomplish the actions and there are no exceptions in both the robotic system and the environment. If the assumption is violated, errors will occur to completely cease the control systems [16][21]. This is because the high-level planning is an open-loop process and it cannot anticipate and also does not consider the unexpected events. This introduces problems and difficulties in handling unexpected events in the robotic system and the environment, such as unexpected obstacles and environment changes.

To address this issue, this paper proposes a planning and control scheme using the perceptive feedback to handle unexpected events after receiving the natural language commands. The perceptive feedback is defined as the feedback representing the perception of the robotic system and the environment. Since the paper is focused on designing the planning and control or the robotic system for natural language control, we assume that the conversion from the natural language commands to the logic representations can always be correctly achieved. Then, a planning and control framework is designed to implement the discrete task scheduling, continuous action planning, and continuous control of the robotic system using the perceptive feedback and make the high-level planning become a closed-loop process. As a result, the system is able to handle some unexpected events happened in the robotic system and the environment.

II. NATURAL LANGUAGE CONTROL OF ROBOTIC OPERATIONS

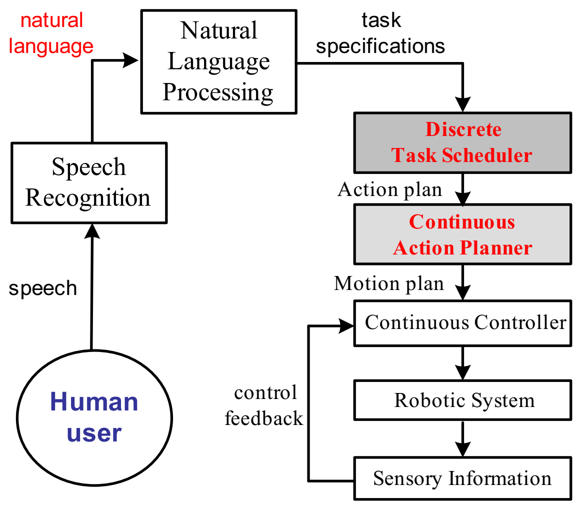


Fig. 1. Natural language control process

The schema of natural language control of robotic operations is shown in Fig. 1. The human user gives commands to the robotic system through speech. The speech is recognized as natural language in text by Voice Recognition. A natural language processing is applied to transform the natural language to formal representations which denote the task specifications. The Discrete Task Scheduler generates the desired discrete behaviors described by actions to realize or satisfy the task specifications. The Continuous Action Planner generates the desired continuous behaviors described by trajectories for the robotic system to achieve the desired actions. The Continuous Controller then controls the robotic system to track the desired trajectories based on feedback of sensory information.

For natural language processing in this paper, a partial parser based on combinatory categorial grammar (CCG) [22] is applied to extract semantic information from human utterances [2]. We have defined a set of basic CCG lexicon rules, which covers key lexicons in our domain, such as actions, object colors, shapes, spatial relations and so forth. Given a human utterance, our CCG parser repeatedly searches for the longest sequence covered by the grammar until the end of the utterance. For example, given the utterance "pick up the small brown block", our parser will generate a semantic representation as follows: `(Action:PICK-UP; Argument:x(size(x,small),color(x,brown), object-type(x, block)))`. Given this representation, the argument x is further grounded to a physical object in the environment based on its semantic constraints. Graph-matching approaches have been developed for this purpose of referential grounding [5][6]. Once the references are grounded, a final state based on the action `PICK-UP` is created. This final state is passed to the task scheduler to come up with a sequence of robotic operations. The translation between actions to final states are currently governed by a set of template rules.

The high-level planning including the task scheduling and action planning and the low-level control work together to realize the natural language tasks. These processes, however, are carried out separately and the task scheduling and action planning are purely open-loop processes. If unexpected events in the robotic system or environment occur, such formalism of control will encounter problems to handle them.

To solve this problem, similar to feedback control, feedback information can be sent to the high-level task scheduling and action planning. In general, the feedback includes the static and dynamic information of the robotic system such as robot position, moving direction, etc., and the environment such as object position, color, shape, weight, etc. It is like the human's perception and thus named *perceptive feedback*. It can be obtained from the sensory information including both robot sensors and environmental sensors.

III. PERCEPTIVE PLANNING AND CONTROL FOR NATURAL LANGUAGE CONTROLLED ROBOTIC SYSTEMS

A. Perceptive Abstracted Model of the Robotic System

Based on the perceptive feedback, the entire system can be abstracted by a finite-state automaton using some predefined rules. In order to obtain a simple model for possible online implementations, only the states and actions (transitions) necessary for the tasks are considered to construct the automaton. The primitive actions are modelled as automata and then connected through the mutual states to construct the final automaton. Each state contains not only a state symbol representing the discrete state but also some variables representing the continuous states such as the positions. Since the model can can be updated based on the perceptive feedback, it is named perceptive system model and described by a 5-tuple automaton

$$\mathrm{A} = (Q(P),\ \Sigma,\ \delta,\ q_0(p_{0,\delta}),\ Q_m(p_{m,\delta})) \tag{1}$$

where

- $Q(P)$ is the set of states determined by the perceptive feedback (Q is the set of state symbols and P is the set of variables). A single state i is represented by $q_i(p_{i,\delta})$ and $p_{i,\delta}$ contains the continuous variables such as positions to represent the initial or final states of all actions related to the state i.
- $\sum$ is the set of actions.
- $\delta : Q(P) \times \sum \rightarrow Q(P)$ is the state transition of actions.
- $q_0(p_{0,\delta}) \in Q(p)$ is the initial state determined by the perceptive feedback.
- $Q_m(p_{m,\delta}) \in Q(P)$ is the set of marked states representing the completion of a task, which is determined by the task specifications and perceptive feedback.

Example 1: Consider that there are three objects and the task is to control a robot to pick up one object. Fig. 2 shows the model to describe the states and transitions of interest for the task. The states are represented by $d_1d_2\,(p_{d_1d_2,\delta})$. The first digit $d_1 \in \{0,1,2,3\}$ means that the end-effector is at the position of object d_1 if $d_1 \in \{1,2,3\}$ and at the initial position if $d_1 = 0$. The second digit $d_2 \in \{0,1\}$ means that the gripper is open if $d_2 = 0$ and closed if $d_2 = 1$. The actions δ are represented by symbols, where m_{ij} means to move the end-effector from position i to position j and o and c means to open and close the gripper respectively. $p_{d_1 d_2,\delta}$ represents the positions of the end-effector or the gripper fingers for actions m_{ij}, o or c related to the state $d_1 d_2$.

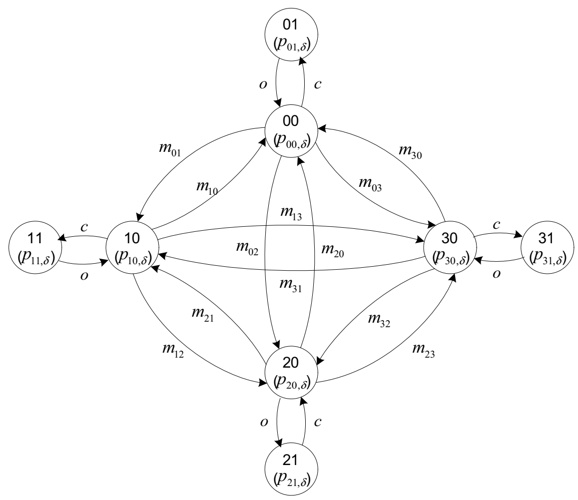


Fig. 2. Example of perceptive system model

B. Perceptive Planning of the Robotic System

The perceptive planner generation is to generate the action and motion plans for the robotic system to realize the given task. It contains three sequential steps. The first step is to generate the discrete solution, i.e., desired action behaviors, which is like the traditional task scheduling. The second step is to generate the continuous solution for the discrete solution, i.e., desired trajectories, which is like the traditional action planning. The third step is to construct a plan to incorporate the discrete and continuous solutions.

In the first step, the task described by natural language can be converted to some formal representations, such as temporal logic formula and automata based on the predefined rules and policies. For a general formulation, the synthesis algorithms in discrete event systems can be adopted to generate the desired action behaviors to realize the task such as the supervisory controller synthesis algorithm [23][24], and linear temporal logic synthesis algorithm [21]. For some specific formulation in natural language control, such as pick-up task in the example 1, the task can be represented by a single marked state. The synthesis is thus to find a solution from the initial state to the marked state in the perceptive system model.

In the second step, based on the variables $p_{i,/delta}$ in the initial and final states of each action, existing online motion planning algorithms can be adopted to generate the desired trajectory for the action. The algorithms include minimum-time based motion planning, minimum-energy based motion planning, etc.

In traditional planning methods, the two steps above have finished the high-level planning. The desired discrete action behaviors and the desired continuous trajectories are then parameterized by time t. They are executed by the low-level controller according to the time evolvement. The high-level planning and low-level control are connected in an open-loop manner, which may cause problems when unexpected events happen. For example, if the robot is stopped, the desired action behaviors and the desired trajectories, however, will still keep evolving since time never stops.

To handle this issue, the third step is to model both action behaviors and motion trajectories using a non-time reference instead of time t. The non-time reference is named perceptive reference and represented by s. It is a variable which can carry the information of the system outputs. We have used this reference to model the continuous trajectories of robotic systems for coordination and teleoperation in our previous works [25][26]. It is extended to model both discrete actions and continuous trajectories in this paper. Different from the method of using multiple references in [27], a single perceptive reference is used for both discrete actions and continuous trajectories. In each action, the perceptive reference s is a piece of continuous variable related to the trajectory, such as travelled distance along the trajectory. The trajectory is then parameterized by s in each action. For multiple actions, the overall perceptive reference is then a piece-wise continuous variable. For the entire plan, similar to the function of time t, the evolvement of s within a piece drives the robot to move along the continuous trajectory and the jump of s between two pieces drives the robot to transit from one action to another action. The perceptive hybrid plan can then be represented by a 6-tuple automaton

$$\mathrm{A}_{PP} = \big(Q(P),\ \Sigma,\ \delta(s),\ Y^d_\delta(s),\ q_0(p_0),\ Q_m(p_m)\big) \quad (2)$$

where $Q(P)$ is the set of states, $\sum$ is the set of events (actions), $\delta(s) : Q(P) \times \sum \to Q(P)$ is the state transition function parameterized by s, $Y^d_\delta(s)$ is the desired continuous trajectory parameterized by s for the action δ, $q_0(p_{0,\delta}) \in Q(p)$ is the initial state, and $Q_m(p_{m,\delta}) \in Q(P)$ is the set of marked states.

Example 2: Take the model in Example 1 as an example. Suppose the initial state is ”01” and the given task is ”Pick up the object 2”. The accomplishment of the given task can be represented by a marked state ”21”. The synthesis solution is then a sequence of actions to drive the state from ”01” to ”21”. Define the perceptive reference s as the distance traveled by the end-effector or the distance between the gripper fingers for different desired actions. The desired trajectories for these actions are planned and then parameterized by s. After this, the perceptive planner is finally obtained and shown in Fig. 3. For each transition δ of an action, s has a specific definition and a valid range $[s^0_\delta, s^f_\delta]$, where s^0_δ denotes the start of the action and s^f_δ denotes the finish of the action. The perceptive reference s is a piece-wise continuous variable. Its values in each piece drive the desired continuous trajectories and its "jumps" between adjacent pieces drive the desired discrete action behaviors.

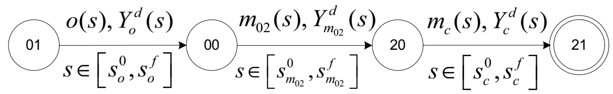


Fig. 3. Perceptive plan example

The solution of high-level planning is represented by a perceptive planner which is parameterized by the perceptive reference that is determined by the perceptive feedback. This creates a mechanism such that the system inputs generated from the high-level planning is determined by the system outputs. Thus, the high-level planning becomes a closed-loop process and is able to handle unexpected events. The way of determining the perceptive reference by the perceptive feedback will be introduced in the following subsection.

C. Perceptive Control of the Robotic System

If there are no unexpected events in the environment, the first step of the perceptive control is to determine the optimal perceptive reference s^* from the real-time perceptive feedback. For a single robot system, to achieve the best trajectory tracking during the execution of an action, a temporary perceptive reference can be computed by minimizing the tracking errors of the robot, which is described by

$$s_{temp} = \underset{s \in S_\delta}{arg\,min} \left\{ \left\| Y_\delta^d(s) - Y \right\| \right\} \tag{3}$$

where $S_\delta = [s_\delta^0, s_\delta^f]$ is the range of perceptive reference during the transition of the action and Y is the real-time output of the robot position. This equation can be solved by orthogonal projection from the current position to the desired trajectory.

When the system lies in the execution of an action, it is driven by continuous behaviors and s_{temp} is in $[s_\delta^0, s_\delta^f)$. Then, the desired continuous behavior should continue to be driven by s_{temp}. When the system just finishes an action and is about to start the next action, s_{temp} is equal to s_δ^f and it will be driven by discrete behaviors. Therefore, the optimal perceptive reference can be designed by

$$s^* = \begin{cases} s_{temp}, & if\ s_{temp} \in \left[s_\delta^0, s_\delta^f\right) \\ s_{next(\delta)}^0, & if\ s_{temp} = s_\delta^f \end{cases} \tag{4}$$

Once the optimal perceptive reference is determined, the inputs for the underlying continuous controller can be obtained by plugging s^* into the perceptive plan.

When unexpected events in the environment, such as moving objects and adding or removing objects, it implies that the previous perceptive system model cannot represent the current system any more and therefore the previously generated system plan is no longer suitable. Then, the perceptive feedback needs to be used to update the perceptive system model and the system plan needs to be regenerated based on the updated system model. Therefore, the general control process is illustrated in Fig. 4 and described as following.

Step 1: Convert the natural language to a formal representation to describe the task specifications, e.g., a marked state;

Step 2: Abstract or update the perceptive system model from the perceptive feedback and generate a perceptive plan to achieve the task specifications;

Step 3: Run the high-frequency perceptive control loop

1) Acquire perceptive feedback from robot and environmental sensors;
2) If unexpected events happen to the environment, return the perceptive feedback to *Step 2*; Otherwise, continue to 3);
3) Generate s^* using the perceptive feedback and use it to determine the instantaneous input $Y_\delta^d(s^*)$ for the continuous controller;
4) Execute continuous controller for achieving $Y_\delta^d(s^*)$;
5) go to 1).

Remarks:

1) If unexpected events happen in the environment, such as moving, adding and removing objects, the perceptive feedback will be used to update the system model and then a new perceptive hybrid plan can be generated to achieve original task. If unexpected events happen in the robotic system, e.g., the robot is unexpectedly stopped, the optimal perceptive reference s^* stops evolving as well. Both the desired discrete action behavior and continuous motion behavior are then suspended. Once the unexpected event is removed, the robot starts to move and then s^* continues evolving. This will automatically recover the desired behaviors with no re-setting or re-planning.
2) The formalism can be extended to natural language control of multiple concurrent robotic systems. The only concern is to design a perceptive reference which can carry perceptive feedback from all concurrent robotic systems. Some ideas of designing the perceptive reference for such systems can be found in [26][28].

IV. Experimental Results for Natural Language Control of Robotic Operations

The experimental setup is shown in Fig. 5. A human user controlled a mobile manipulator to operate three blocks (a big red block, a big brown block, and a small brown block) using speech. The speech was recognized as natural language text using the Dragon Speech Recognition software and converted to a formal language "action+object" using the semantic processing. It was then mapped to a formal representation, i.e., a marked state, based on predefined mapping rules. The perceptive feedback contains the positions of the end-effector and gripper fingers and the positions, color and shape of the objects which are obtained from a calibrated vision system.

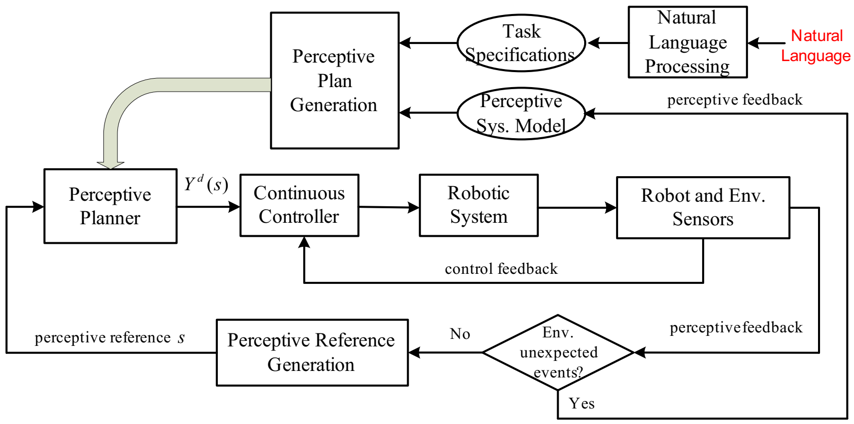


Fig. 4. Natural language control using the perceptive feedback

In Fig. 6 shows the result of handling unexpected events in the robotic system. The human user asked the robot to pick up the big red block by saying ”Pick up the red block”. While the robot was moving to the red block, at 7 s and 12 s, it was blocked by a human’s hand. Then, the perceptive reference stopped evolving, which suspended all following desired discrete action behaviors and desired continuous trajectories. When the hand was removed at 11 s and 14 s, the perceptive reference started to evolve again and automatically drove the robot to accomplish the following action behaviors and trajectories. The task was then finally finished.

Fig. 7 and 8 show the results of handling unexpected events in the environment. The human user asked the robot to pick up the big brown block and big red block by saying ”Take that big brown block” and ”Grasp that red block” respectively. While the robot was moving to them, the big brown block was moved to other positions twice at 11 s and 22 s in Fig. 3 and the red block was swapped with the big brown block at 4 s in Fig. 9. The perceptive feedback was then sent back to the update the perceptive system model and new plans were also generated to achieve the original tasks.

Although the designed tasks are simple, these results can clearly demonstrate the advantages of the designed mehod for handling unexpected events in the robotic system and the environment.

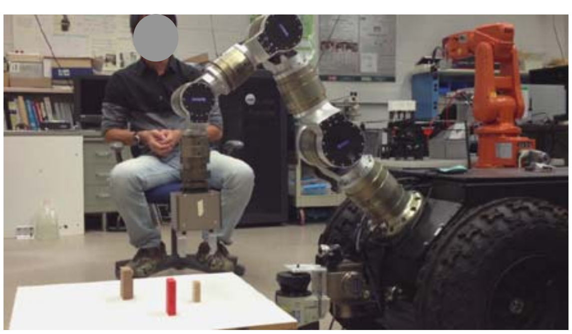


Fig. 5. Experimental setup

V. CONCLUSIONS

A new natural language control scheme for robotic operations using the perceptive feedback has been proposed. The high-level planning including discrete task scheduling and continuous action planning and the low-level continuous control are presented. The high-level planning solution is driven by a perceptive reference which is determined by the perceptive feedback. This makes the high-level planning become closed-loop such that the robot can accomplish the high-level natural language tasks in the presence of unexpected events in both the robotic system and the environment. Although the examples in the paper are simple, the proposed scheme provides a framework for designing complicated tasks for natural language controlled robotic systems in handling more unexpected events in our future work. In addition, using the perceptive feedback to assist the natural language processing for converting the natural language commands to the logic representations will also be our future work.

REFERENCES

[1] C. Breazeal, A. Brooks, J. Gray, G. Hoffman, C. Kidd, H. Lee, J. Lieberman, A. Lockerd and D. Mulanda, ”Humanoid robots as cooperative partners for people,” Journal of Humanoid Robots, 1, 2004.

[2] J.Y. Chai, L. She, R. Fang, S. Ottarson, C. Littley, C. Liu and K. Hanson, ”Collaborative effort towards common ground in situated human robot dialogue,” 9th ACM/IEEE International Conference on Human-Robot Interaction, Bielefeld, Germany, 2014.

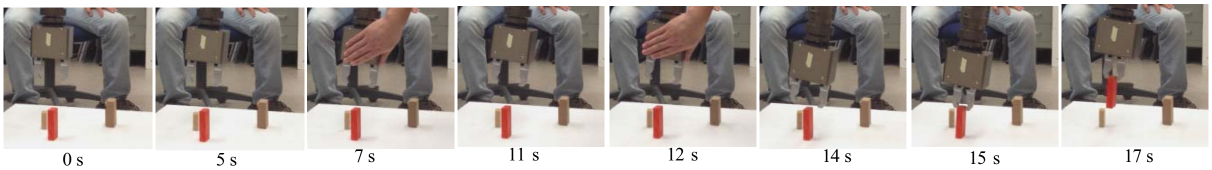
[3] H.I. Christensen, G.M. Kruijff and J. Wyatt, Cognitive Systems, Springer, 2010.

[4] R. Fang, C. Liu and J.Y. Chai, ”Integrating word acquisition and refer- ential grounding towards physical world interaction,” 14th ACM International Conference on Multimodal Interaction, ICMI ’12, 2012, pp. 109-116.

[5] C. Liu, R. Fang and J.Y. Chai, ”Towards mediating shared perceptual basis in situated dialogue,” Proceedings of the SIGDIAL 2012 Conference, 2012, pp. 140-149,.

[6] C. Liu, R. Fang, L. She and J.Y. Chai, ”Modeling collaborative referring for situated referential grounding,” Proceedings of the SIGDIAL 2013 Conference, Metz, France, 2013, pp. 78-86.

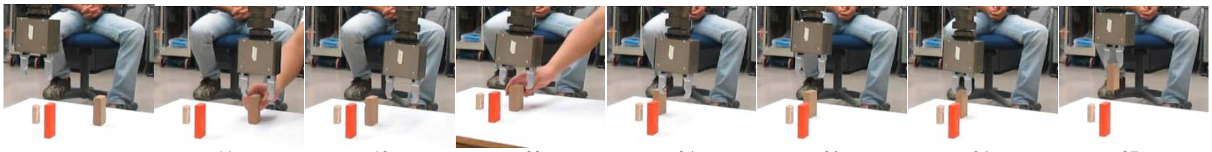
[7] D. Roy, ”Grounding words in perception and action: computational insights,” TRENDS in Cognitive Sciences, vol. 9, no. 8, pp. 389-396, 2005.



0 s 5 s 7 s 11 s 12 s 14 s 15 s 17 s

“Pick up that red block”

Fig. 6. Results of natural language control: the robot is blocked



0 s 11 s 13 s 22 s 24 s 30 s 34 s 37 s

“Take the big brown block”

Fig. 7. Results of natural language control: the object is moved

0 s 3 s 4 s 6 s 12 s 14 s 19 s

“Grasp that red block”

Fig. 8. Results of natural language control: the object is swapped

[8] S. Lauria, T. Kyriacou, G. Bugmann, J. Bos, and E. Klein, ”Converting natural language route instructions into robot-executable procedures,” in Proceedings of the 2002 IEEEInternational Workshop on Robot and Human Interactive Communication, 2002, pp. 223-228.

[9] A. J. Martignoni III and W. D. Smart, ”Programming robots using high-level task descriptions,” in Proceedings of the AAAI Workshop on Supervisory Control of Learning and Adaptive Systems, 2004, pp. 49-54.

[10] M. Nicolescu and M. J. Mataric, ”Learning and interacting in human-robot domains,” IEEE Transactions on Systems, Man, and Cybernetics, Part B: special issue on Socially Intelligent Agents-The Human in the Loop, vol. 31, no. 5, pp. 419-430, 2001.

[11] M. Kloetzer, C. Belta, ”Temporal Logic Planning and Control of Robotic Swarms by Hierarchical Abstractions,” IEEE Transactions on Robotics, vol. 23, no. 2, pp. 320-330, 2007.

[12] C. Liu, J. Walker, and J. Y. Chai, ”Ambiguities in Spatial Language Understanding in Situated Human Robot,” AAAI 2010 Fall Symposium on Dialogue with Robots, 2010.

[13] M. Skubic, D. Perzanowski, S. Blisard, A. Schultz, W. Adams, M. Bugajska, D. Brock, ”Spatial language for human-robot dialogs,” IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews, vol. 34, no. 2, pp. 154-167, 2004.

[14] S. Konrad and B. H. C. Cheng, ”Facilitating the construction of specification pattern-based properties,” in Proceedings of the IEEE International Requirements Engineering Conference, 2005, pp. 329-338.

[15] E. A. Topp, H. Huttenrauch, H. I. Christensen, and K. S. Eklundh, ”Bringing together human and robotics environmental representations-a pilot study,” in Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, 2006, pp. 4946-4952.

[16] H. Kress-Gazit, G. E. Fainekos and G. J. Pappas, ”Temporal Logic-based Reactive Mission and Motion Planning,” IEEE Transactions on Robotics, vol. 25, no. 6, pp. 1370-1381, 2009.

[17] W. Takano and Y. Nakamura, ”Statistically integrated semiotics that enables mutual inference between linguistic and behavioral symbols for humanoid robots,” in Proceedings of the IEEE International Conference on Robotics and Automation, 2009, pp. 646-652.

[18] W. Takano and Y. Nakamura, ”Bigram-based natural language model and statistical motion symbol model for scalable language of humanoid robots ,” in Proceedings of the IEEE International Conference on Robotics and Automation, pp. 1231-1237, 2012.

[19] M. Ralph, M. A. Moussa, ”Toward a Natural Language Interface for Transferring Grasping Skills to Robots,” IEEE Transactions on Robotics, vol. 24, no. 2, pp. 468-475, 2008.

[20] X. He, T. Ogura, A. Satou, O. Hasegawa, ”Developmental Word Acquisition and Grammar Learning by Humanoid Robots Through a Self-Organizing Incremental Neural Network,” IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics, vol. 37, no. 5, pp. 1357-1372, 2007.

[21] H. Kress-Gazit, G. E. Fainekos and G. J. Pappas, ”Translating structured English to robotic controllers,” Advanced robotics, no. 22, pp. 1343- 1359, 2008.

[22] M. Steedman and J. Baldridge, ”Combinatory categorial grammar,” Non-Transformational Syntax, Oxford: Blackwell, pp. 181-224, 2011.

[23] P. J. Ramadge and W. M. Wonham, ”The control of discrete event systems,” Proceedings of the IEEE, vol. 77, no. 1, pp. 81-98, 1989.

[24] J. Goryca, R.C. Hill, ”Formal synthesis of supervisory control software for multiple robot systems,” in Proceedings of American Control Conference, 2013, pp.125-131.

[25] N. Xi, T.J. Tarn and A.K. Bejczy, ”Intelligent Planning and Control for Multirobot coordination: An Event-Based Approach,” IEEE Transaction On Robotics and Automation, vol. 12, no. 3, pp. 439-452, 1996.

[26] Y. Jia, N. Xi and J. Buether, ”Design of single-operator-multi-robot teleoperation systems with random communication delay,” in Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, 2011, pp. 171-176.

[27] Y. Sun, N. Xi and Y. Wang, ”Modeling and analysis of perceptive robot controller based on hybrid automata,” IEEE International Conference on Robotics and Automation, 2004, pp. 2924-2929.

[28] Y. Jia and N. Xi, ”Coordinated Formation Control for Multi-Robot Systems with Communication Constraints,” IEEE/ASME International Conference on Advanced Intelligent Mechatronics, 2011, pp. 158-163.