A Syntactic Approach to Robot Learning of Human Tasks from Demonstrations

Kyuhwa Lee

Submitted in part fulfilment of
the requirements for the degree of
Doctor of Philosophy

Department of Electrical and Electronic Engineering
Imperial College of Science, Technology and Medicine

Declaration

I herewith certify that all material in this dissertation which is not my own work has been properly acknowledged.

Kyuhwa Lee

The copyright of this thesis rests with the author and is made available under a Creative Commons Attribution-Non Commercial-No Derivatives licence. Researchers are free to copy, distribute or transmit the thesis on the condition that they attribute it, that they do not use it for commercial purposes and that they do not alter, transform or build upon it. For any reuse or distribution, researchers must make clear to others the licence terms of this work.

To my family.

Abstract

The successful development of general-purpose humanoid robots, in contrast to traditional pre-programmed problem solving machines, has opened a new research area of how a robot could be programmed by an end-user, not engineers, to suit individual needs.

In this respect, Robot Learning from Demonstration has been actively studied, aiming to enable robots learn various tasks from human users. Although much effort has been put, there are many challenges still remaining until the goal is realized. One of the important challenges is the automatic learning of task representations and reuse of the learned tasks, where each task can be expressed as a series of primitive action components. To deal with such challenges, syntactic approaches to task learning and related issues are investigated.

Firstly, efficient goal-oriented task representation methods using stochastic context-free grammars are studied, which enable robots to understand the human's intended actions even in the presence of both observation errors and human execution errors. By exploiting the task knowledge, it is demonstrated that the robot can correctly identify unexpected, out-of-context actions and perform

the intended actions under reasonable amount of noise. Taking a step further, the automatic learning of these task representations from human demonstrations are studied. It is demonstrated throughout the experiments that the robot is able to learn critical task structures and generalize them. This is essential for understanding more complex tasks sharing the same underlying structures. Following these studies, an unsupervised discovery of the optimal set of primitive action detectors required to represent a task is studied.

Through a diverse set of real-world and simulated experiments that include learning object-related games, postural sequence tasks of dance and surveillance tasks, this thesis demonstrates the effectiveness of syntactic approaches for robot learning from demonstrations.

Acknowledgments

I have been greatly privileged to work with my supervisor, Dr. Yiannis Demiris, who has been an excellent academic adviser as well as personal adviser during my PhD studies. Yiannis has been incredibly patient and wise on guiding my research throughout countless discussions, making sure that I focus on the important topic instead of getting distracted to too many seemingly interesting topics, while providing a lot of opportunities for meeting with many other prominent researchers around the world.

I was also very fortunate enough to have the opportunity to work with Dr. Tae-Kyun Kim, who has substantially influenced my research. His deep knowledge in computer vision and machine learning and critical way of thinking were essential on developing my work.

I also acknowledge Dr. Angelo Cangelosi, who has provided the invaluable RobotDoc Collegium experience over 4 years, a multi-national robotics doctoral training network. It has undoubtedly broadened my research horizon and inspired me at different stages of my education.

My PhD experience would have been incomplete without the teaching experience I gained from the exceptional course director, Mr. Liam Madden. Liam consistently researched on effective teaching methods using various state-of-the-art technologies, which I was able to learn while closely working with him.

I sincerely thank my family who have been always loving, supporting and trusting me throughout the studies. My PhD would have been impossible without them.

I thank our members of Personal Robotics Lab who are excellent students and post-docs: Dimitri Ognibene, Yanyu Su, Arturo Ribes, Harold Soh, Miguel Sarabia, Tom Carlson, Yan Wu, Simon Butler, Sotirios Chatzis, Eris Chinellato, Dimitrios Korkinof, Raquel Ros Espinoza, Ayşe Küçükyılmaz, Alexandre Coninx, Yixing Gao, Theodosis Georgiou, Nicola Catenacci, Jonathan Classens, Bálint Takács, Paschalis Veskos and Murilo Fernandes Martins.

Contents

1. Introduction **21**

1.1. Motivation 22

1.2. Thesis Summary 23

1.3. Contributions 24

1.4. Roadmap 25

1.5. List of resulting publications 27

2. Background and Related Work **28**

2.1. Introduction 28

2.2. Robot Learning from Demonstrations 28

2.3. Task Representation and Recognition 30

2.3.1. Task Representation through SCFG 32

2.3.2. Task Recognition through Probabilistic Parsing 34

2.4. Task Structure Learning 37

2.5. Task Structure for Attention Control 39

2.6. Summary 42

3. Syntactic Approaches to Task Representation and Recognition **43**

3.1. Introduction 43

3.2. Experiments 43
3.2.1. Experiment Design 45
3.2.2. Findings 53
3.3. Summary 57

4. Learning Task Structures from Demonstrations 60
4.1. Introduction 60
4.2. The Discovery of Task Structures and Parameters 62
4.2.1. Active Substring Discovery 64
4.2.2. Considering Input Samples with Uncertainty 66
4.2.3. Measuring the Quality of a Grammar 69
4.3. Bag-of-Balls Experiment 70
4.3.1. Experiment Design 70
4.3.2. Findings 73
4.4. The Towers of Hanoi Experiment 74
4.4.1. Experiment Design 75
4.4.2. Findings 78
4.5. The Dance Imitation Experiment 83
4.5.1. Experiment Design 83
4.5.2. Findings 88
4.6. The Effect of Pruning Factors 92
4.7. Summary 94

5. Learning Action Components from Demonstrations 97
5.1. Introduction 97
5.2. Automatic Discovery of Primitive Action Detectors 97
5.2.1. Discovery of Candidate Symbols 99
5.2.2. Selecting the Number of Symbols 100

5.3. Experiments . . . 102
5.3.1. Experiment Design . . . 102
5.3.2. Findings . . . 104
5.4. Summary . . . 109

6. Action Anticipation and Attention Allocation using Task Structures 111

6.1. Introduction . . . 111
6.2. Using Task Structure Information for Action Anticipation . . 113
6.3. Experiments . . . 118
6.3.1. Experiment Design . . . 118
6.3.2. Findings . . . 119
6.4. Summary . . . 126

7. Conclusions and Future Work 127

7.1. Conclusions . . . 127
7.2. Open Questions and Future Work . . . 129

Bibliography 130

Appendix A. Experimental Setup 147

A.1. Robot Platform . . . 147
A.2. Motion Capture System . . . 148

List of Figures

2.1. A simple grammar that represents a task where a robot enters one or more rooms until "meet" event is happened, and returns to the initial location. 33

3.1. Building problem-dependent task representations from a pool of generic task templates. Primitive action detectors are trained offline to convert the input signals into the time-series terminal symbols for the SCFG parser. By observing a human demonstration and parsing the observation, the system classifies the demonstration into a corresponding task and assigns the task property, e.g. 1, 2, ... 4. 44

3.2. The procedure of the experiment. 46

3.3. Software modules used in the experiment. 47

3.4. Task templates used in the experiment. Three different types of tasks are shown in the top part (G_{DROP}, G_{PLACE}, $G_{NEXTBOX}$), followed by non-terminal symbols commonly shared across three grammars. 49

3.5. Examples of object segmentation. Images are acquired from the both eyes of iCub, from which depth map is computed and blobs closer than a threshold are segmented. 51

3.6. Extracted patches and their color histograms. In histogram images, x-axis represents the color bin and y-axis represents the frequency. Finger colors in the patch are suppressed for better tracking performance. 51
3.7. iCub observing the object organization task demonstrated by a human participant. 54
3.8. iCub performing task by executing parsed primitive actions. . 54
3.9. Sample terminal symbols generated by primitive action detectors and the parsed result. The action symbols with the highest likelihood are underlined. The likelihood values of aobj in time steps 2 and 3 are unexpected, which are corrected after concluding that the demonstration was a DROP task. For naming conventions, please refer to Figure 3.4. . . . 56
3.10. N:Nextbox, D:Drop, P:Place, X:Recognition Failure 57
3.11. Confusion Matrix 57

4.1. Overview of the approach to task learning with an example scenario. The input training sequences are converted into streams of symbols with probability, respectively indicated by circles and numbers below, from which the original structure is uncovered using grammatical representations. The acquired knowledge is used to better recognize unforeseen, more complex tasks (test sequences) that share the same structure components. 61

4.2. (a) Initial naive grammar. (b) After *Substituting* AB with X, AC with Y, and XX with Z. (c) After *Merging* (X, Y) to X. (d) After *Merging* (X, Z) to Z. (e) After *Merging* (S, Z) to S. Please note that uncertainties of symbols are not considered in this example. 64

4.3. Learning summary 71

4.4. Description length ratios of grammars generated by different methods. The lower score indicates that the grammar is more compact yet maintains sufficient expressive power. 72

4.5. Actual MDL scores for each method compared with the model grammar. MDL scores are averaged over 10 trials for each noise condition. The graph is shown with a 2% step for better view. A lower score indicates that the grammar is more compact yet reasonably expressive. How these scores affect the performance in the real world will be discussed in Sections 4.4 and 4.5. 72

4.6. The obtained grammars using the method in (Kitani, Yoichi, and Sugimoto, 2008)(a) and the proposed method(b) from data with noise probability 0.08. 73

4.7. (a-b) A sample tracking screen while a human participant is solving the puzzle with 4 disks. Compared to the low-noise condition (a), the high-noise condition (b) shows overexposed spots which often makes the tracker unstable. The tracker immediately resets the position if lost by searching the desired blob from the entire region of the image. (c) shows iCub performing parsed primitive actions. A demo video is available at: http://www.youtube.com/watch?v=S99ViThK050 . 76

4.8. Primitive actions defined in *Towers of Hanoi* experiment. The system is equipped with these 5 primitive action detectors which generates symbol probability during observation. . 77
4.9. Success rates using our method, base method and the pure imitation. Scenarios LL and LH: Train on the low-noise condition and test on low- and high-noise conditions, respectively. Scenarios HL and HH: Train on high-noise condition and test on low- and high-noise conditions, respectively. The fact that a single mistake while parsing a long test sequence causes a failure makes this problem non-trivial. 78
4.10. Detailed results with average MDL scores for comparison. Each case is tested on 75 sequences. MDL score is not available for the pure imitation as it does not rely on any learned model. It is worth noting that lower MDL scores generally lead to higher success rates. 79
4.11. Error statistics of demonstrations using 4 disks on each noise condition. Note that even in the low-noise condition, there are only 5 trials observed with all correct symbols, which means that in most cases the pure imitation will not lead to the desired goal state. Each testing sequence is composed of 45 primitive action symbols, which makes this problem non-trivial as only a single mistake will make it fail to achieve the goal. 79
4.12. A test experiment scenario where 4 disks are used, requiring 45 actions to be correctly executed to reach the goal. 81

4.13. (a) A sample grammar that captured the meaningful task components such as LAD, LBD, and LCD, which can be used to enforce the observation to be consistent with the demonstrator's intended actions. $CADSS$ and $SLBAS$ come from occasional noisy examples and hence they are assigned very probabilities. (b) A grammar learned from an ideal (noise-free) dataset. (c) A grammar learned from the same dataset of (a), but with a pruning threshold of 0.15. Please see Section 4.6 for more detailed analysis on pruning effects. . 82

4.14. 9 motion primitives used in this experiment and a demonstration example. Please see the following video for better visualization: http://www.youtube.com/watch?v=S99ViThK050. 84

4.15. 3 types of dance representations used in the experiment. Please see Figure 4.14 for reference. In training set, there are 5 trials for each value of n (sequence length), which results in 40 dance demonstrations (225 input symbols). The testing set has 6 trials for each n, which results in 36 dance demonstrations (450 input symbols). 85

4.16. The informative human joints chosen to be used for calculating the ASV and ASD values. As these joints are often overlapped across multiple motion primitives, the number of the joint sets are reduced to four. 85

4.17. The ASV(a) and ASD(b) of the movement sequence: the used joints set for each time step is marked on the bottom using corresponding color. The zero-crossings of ASV with sufficiently low ASD value are chosen as the segmentation points. 87

4.18. iCub performing parsed primitive actions. Each figure from the left to right respectively represents primitive actions C, D, E, and F. The video containing full movements can be seen on: http://www.youtube.com/watch?v=S99ViThK050. 89
4.19. Detailed results with average MDL scores for comparison. Each scenario has 36 sequences, and the total number of symbols per scenario is 450. "Correct" column shows the number of correctly recognized symbols after parsing, where in pure imitation case it is equivalent to the number of action detector errors. MDL score is not available for the pure imitation as it does not rely on any learned model. It can be seen that the lower MDL scores generally lead to higher success rates. . 90
4.20. Acquired grammars from automatically segmented dataset using the method described in Section 4.5.1. The error in the segmentation leads to a higher error rate on detectors, which is regarded as the high-noise scenario. 91
4.21. Learned grammars from manually segmented dataset, noted as the low-noise scenario. Note that only segmentation was done manually, where symbol detectors are still trained and tested in the same way as in automatically-segmented dataset. 91

4.22. The effect of different pruning parameters. In this experiment, we trained from all training data, i.e. all samples from both low-noise and high-noise conditions, and similarly tested on all testing samples. It can be seen that although overall MDL score decreases as threshold increases, the resulting grammar loses generality and shows poor performance on testing data. As in the previous experiments, a trial is regarded as fail even if there was a single error in parsed symbols. 93

4.23. The comparison of training times over different prune parameters. Since rules are more likely to be pruned as the threshold increases, the overall learning time tends to decrease. It was tested on a Linux desktop with i7 3.2GHz CPU, 16GB RAM, Python 3.2. 93

5.1. Overview: Candidate symbols are generated using agglomerative hierarchical clustering approach, where too general or specific symbols are subsequently filtered out by measuring the model complexity and likelihood. 98

5.2. Summary of action symbol selection. 103

5.3. An example clustering tree created (top), showing only the top 30 nodes for better view, and eight action symbol representations (bottom) obtained from the Towers of Hanoi dataset. 105

5.4. The spanning values of description lengths obtained from the Towers of Hanoi (top) and Dance (bottom) data. Best cases (S^*) obtained using the method described in Sec. 5.2.2 are indicated by square markers. (Best viewed in color.) 106

5.5. Results on the *Towers of Hanoi* (**T**) and *Dance* (**C**) dataset. α and β denote mean $\pm$ standard deviation of $-logP(M)$ and $-logP(D|M)$, respectively. Votes (**V**) are computed by the method described in Section 5.2.2, whereas success rates (**S**) are computed by comparing the parsed symbols. 107
5.6. Representative visualized snapshots of the Dance dataset. . . 108
5.7. Example grammars learned from data. (a) A grammar generated by a system ψ_6 having 6 symbols A-F. (b) has 1 less symbol, where one of the symbols represents two different actions. (c) has 1 more symbol, where the same action could be represented with two different symbols. Low-probability rules ($< 3\%$) exist due to input data noise. 108

6.1. Single vs Random event. Circles and squares denote the attended point when MEA and MMIA were used, respectively. With MEA, window 1 is favored from t=7, whereas with MMIA the system loses interest on window 1 and starts exploring window 2. As a result, window 1 is watched only 4 times, compared to 10 with MEA, without losing too much information that is required to recognize the event happened inside. 120
6.2. Two concurrent events. Circles and squares denote the attended point when MEA and MMIA were used, respectively. 121

6.3. Example *Delivery* task scenario. The blue and red boxes show window attended using MMIA and MEA, respectively. The bottom left and right graphs show the expected entropy of windows under MEA policy, and the mutual information of windows under MMIA policy, respectively. (VIRAT-000006, frames 14648-16277) 124

6.4. ROC curves obtained under different attention policies and their respective ROC area values. 125

A.1. iCub performing tasks demonstrated by human partners. . . 148

1. Introduction

Since the realization of the technological advancement in humanoid robotics, it has become a desirable property for a humanoid robot to be capable of performing more human-like interactions with human users who are not essentially robot experts. Also commonly known as Imitation Learning, the Learning from Demonstrations (LfD) methods allow a human user to add new capabilities to the robot in an intuitive manner without explicit re-programming. A large amount of research has been conducted in this domain in recent years from the motor-level trajectory learning to the symbolic-level task structure learning, yielding various types of learning algorithms and knowledge representations. This thesis investigates the problems involved in making a robot to learn and imitate structured human tasks by integrating syntactic methods into the LfD paradigm.

The goal of the motor-level trajectory learning is to learn a human-like continuous movement of an action, while the symbolic-level task structure learning aims to learn the relationship between these multiple actions. Compared to trajectory learning, task structure learning has been relatively less explored, although it has a significant impact in many human-robot interaction areas, such as action anticipation, human intention detection, attention control, turn taking and assistive robotics. Syntactic representations are well suited for these problems as they provide an efficient and intuitive

way of representing a task as a composite of actions. This thesis presents a computational model of learning task representations from human demonstrations using syntactic approaches and demonstrates its effectiveness on various real-world scenarios.

1.1. Motivation

Humans are capable of learning novel activity representations despite the noisy sensory input by making use of the previously acquired contextual knowledge, since many human activities often share the similar underlying structures. For example, when we observe someone's hand transferring an object to another place, where the grasping action cannot be seen due to some occlusions, we can still infer that a grasping action occurred before the object was lifted. From this analogy, if a robot has knowledge about a minimal set of basic actions which are frequently used in human-robot interaction environments, it can boost the performance of learning new tasks. The use of such knowledge enables a learner to incrementally acquire a new knowledge without the need of excessive verification processes, resulting in a more natural interaction.

In the real-world environment, there are still many obstacles yet to be solved for a robot to be successful in imitation learning. One of them is dealing with the limited capability of sensors of robotic systems in real-world environment, such as noise and occlusions. It is often desirable to minimize the sensory input error to allocate more resources on solving task-level problems.

To realize this idea as a formal computational model of the human task learning mechanism, the following requirements need to be satisfied:

- It should be able to represent the structure of a task in an intuitive and efficient way.
- It should be able to automatically extract the underlying task structures which may be hierarchical and recursive.
- It should be able to cope with observation errors as well as human demonstration errors, which are inherent in many real-world scenarios.

This thesis aims to address these challenges by augmenting the LfD approaches with a robust task representation framework based on stochastic context-free grammars (SCFG), which is an effective tool for defining the semantic constraints of a task.

1.2. Thesis Summary

The Learning from Demonstrations paradigm integrated with syntactic methods allows intuitive and flexible task representations while providing mechanisms to automatically recognize and extract important task structures from human users, as well as the execution of actions. This thesis provides how the LfD paradigm can be realized with a syntactic approach using efficient task representation methods, followed by the robust recognition of these tasks from observation considering the uncertainty of the sensory input. Subsequently, the automatic learning of these tasks from human demonstrations is studied, which is further developed to discover a set of primitive action detectors that can be used as the building blocks of tasks. Finally, the possibility of predicting actions online based on the learned task structure during the observation is investigated.

1.3. Contributions

This thesis offers the following contributions:

(1) It proposes a novel approach to the adaption of the stochastic context-free grammars (SCFG) framework into the LfD paradigm in three important areas: task recognition, task learning and task execution.

(2) It provides experimental findings (Chapters 3.2.2 and 4.3-4.5) while using SCFG as a task representation framework throughout multiple real-world and simulated tasks including games, dance and surveillance. Various action detection methods for generating symbols with confidence values on different types of input signals are demonstrated and the related issues while observing human movements are discussed. The dataset used include vision datasets obtained from cameras, 3-D point cloud datasets obtained from a motion capture system and simulated datasets.

(3) A computational model of the structured human task learning is developed that automatically discovers and extracts the important aspects of task structures in the form of SCFG.

(4) It addresses effective methods to deal with observation errors and human demonstration errors that occur while training and testing the robot by explicitly taking into account the uncertainties inherent in human action detectors. The effect of the grammar rule pruning factor, an important factor while learning task grammars, is systematically tested and the results are compared in terms of learning time, model complexity and accuracy. It is also experimentally shown that the quality score of a learned grammar, which is at its best when the model complexity and the model accuracy are perfectly balanced, coincides well with the expected theoretical results.

(5) As a generalization to the above method, an automatic learning ap-

proach to discovering the optimal set of primitive action symbols for efficiently describing a task is developed and the experimental findings on two different datasets are reported.

1.4. Roadmap

The rest of the thesis is organized as follows:

Chapter 2 - Background and Related Work describes the related research and background information about other LfD approaches and syntactic models. Furthermore, techniques commonly used for detecting actions and their applications are also presented.

Chapter 3 - Syntactic Approaches to Task Representation and Recognition presents the utility of SCFG-based task representation methods as a tool for defining task templates and demonstrates how it can be used to infer the intended action of the human demonstrator under noisy observation settings.

Chapter 4 - Learning Task Structures from Demonstrations presents the automatic learning of task structures and parameters from human demonstrations. The learning effect under different noise conditions is investigated as well as different pruning factors.

Chapter 5 - Learning Action Components from Demonstrations further presents the automatic learning of action detectors to optimally represent a task by utilizing the method described in Chapter 4.

Chapter 6 - Action Anticipation and Attention Allocation using Task Structures presents a step towards dynamic attention control system for efficient long-term task recognition. The structured representations of tasks are exploited to actively decide not only *where*, but also *when* to

retrieve information to maximally improve the recognition of task activities given bounded computational resources.

Chapter 7 - Conclusion summarizes the key points of this thesis and presents possible future extensions of this research.

1.5. List of resulting publications

- **Kyuhwa Lee, Yanyu Su, Tae-Kyun Kim and Yiannis Demiris**: "A Syntactic Approach to Robot Imitation Learning using Probabilistic Activity Grammars," *Robotics and Autonomous Systems*, Elsevier, Volume 61, Issue 12, pp.1323-1334, 2013. (Chapter 4)

- **Kyuhwa Lee, Tae-Kyun Kim and Yiannis Demiris**: "Learning Reusable Task Components using Hierarchical Activity Grammars with Uncertainties," *IEEE International Conference on Robotics and Automation*, pp.1994-1999, 2012. (Chapter 4)

- **Kyuhwa Lee, Tae-Kyun Kim and Yiannis Demiris**: "Learning Action Symbols for Hierarchical Grammar Induction," *The 21st International Conference on Pattern Recognition*, pp.3778-3782, 2012. (Chapter 5)

- **Kyuhwa Lee and Yiannis Demiris**: "Towards Incremental Learning of Task-dependent Actions using Probabilistic Parsing," *IEEE International Conference on Development and Learning*, pp.1-6, 2011. (Chapter 3)

- **Yanyu Su, Yan Wu, Kyuhwa Lee, Zhijiang Du, Yiannis Demiris**: "Robust Grasping for an Under-actuated Anthropomorphic Hand under Object Position Uncertainty," *IEEE-RAS International Conference on Humanoid Robots*, pp.719-725, 2012. (Chapter 4)

- **D. Ognibene, Y. Wu, K. Lee, and Y. Demiris**: "Hierarchies in Embodied Action Perception," in *Computational and Robotic Models of the Hierarchical Organisation of Behaviour, G. Baldassare and M. Mirolli (eds)*, Springer Verlag, pp.81-98, 2012. (Chapters 3 and 4)

2. Background and Related Work

2.1. Introduction

This chapter first presents the prior work in the Learning from Demonstrations (LfD) domain with the examination of various issues related to this research field. It will then move on to the topic of human task representation methods that are essential for expressing the learned task knowledge. Prior work of using syntactic approaches on modeling human behaviors will be reviewed, followed by algorithmic issues related to stochastic context-free grammars which will be used as basis for representing tasks in this thesis.

2.2. Robot Learning from Demonstrations

There has been a growing interest in developing autonomous robots which are capable of learning goal-directed actions by imitating humans using multi-level representations of actions (Kuniyoshi, Inaba, and Inoue, 1994; Pardowitz, Knoop, Dillmann, and Zollner, 2007; Argall, Chernova, Veloso, and Browning, 2009).The LfD has been widely studied over the past decade with the aim of providing an efficient means of teaching tasks to robots (Argall, Chernova, Veloso, and Browning, 2009; Billard, Calinon, Dillmann, and Schaal, 2008; Asada, Ogino, Matsuyama, and Ooga, 2006; Dillmann, 2004; Schaal, 1999; Kuniyoshi, Inaba, and Inoue, 1994). Instead of explicitly

programming the required sequence of actions, it is intended that human users teach robots in a more natural way. Achieving this capability is, of course, quite challenging (as discussed in (Breazeal and Scassellati, 2001; Breazeal and Scassellati, 2002; Thrun and Mitchell, 1995)), and Dautenhan and Nehaniv (Dautenhahn and Nehaniv, 2002) classify the problem into the following domains: *who* to imitate, *when* to imitate, *how* to imitate, *what* to imitate, and *how to judge* if an imitation was successful.

It is known that humans tend to interpret actions based on goals rather than motion trajectories (Baldwin and Baird, 2001; Woodward, Sommerville, and Guajardo, 2001) and this thesis frames the problems of LfD in this perspective. More emphasis is put on the issue of *what* to imitate, where a robot does not intend to copy the exact trajectories of actions, but to deduce the intention of the demonstrator. As stated in (Jansen and Belpaeme, 2006), this requires a different approach to solving the problems of *how* to imitate, e.g. (Billard, Epars, Calinon, Schaal, and Cheng, 2004; Billard, 2001; Erlhagen, Mukovskiy, Bicho, Panin, Kiss, Knoll, Schie, and Bekkering, 2006; Wu and Demiris, 2010; Nguyen-tuong and Peters, 2008; Gurbuz, Shimizu, and Cheng, 2005; Soh, Su, and Demiris, 2012), where more emphasis is put on the imitation of observed motion trajectories using robot's own capabilities in continuous time domain. In (Lockerd and Breazeal, 2004; Ekvall and Kragic, 2008; Bentivegna, Atkeson, and Cheng, 2006; Calinon, Guenter, and Billard, 2005; Demiris and Hayes, 2002; Lee, Lee, Thomaz, and Bobick, 2009; Chao, Cakmak, and Thomaz, 2011), various types of tasks are defined to include action sequences with some degree of recursion, but they differ from the main contribution of this thesis as hierarchically structured tasks are not considered.

2.3. Task Representation and Recognition

The task representation is an important issue in the LfD framework. There is a vast amount of work done on human task representation from computer vision and machine learning communities, as summarized in (Aggarwal and Ryoo, 2011). Language-inspired human behavior representation is one of the important subjects in the research domains of autonomous robots and robot intelligence (Cangelosi et al., 2010; Dominey and Boucher, 2005; Dominey, 2002; Cangelosi and Parisi, 2002; Petit et al., 2013). As this thesis mainly focuses on the task-level learning, syntactic models such as stochastic context-free grammars (SCFG) and hidden Markov models (HMM) are used. SCFG is essentially a stochastic model that extends context-free grammar similar to HMM which extends regular grammars, but with stronger expressive power. SCFG are particularly well suited for our purpose due to its easiness on representing hierarchies of actions and robustness against observation noise. Advantages on using SCFG model in the LfD framework are summarized as follows:

First, it can utilize syntactic knowledge instead of relying on pure statistics to solve a problem as they can be expressed. Second, it can clarify ambiguous actions detected at the sensory level during the parsing process. Once the parsing is done, the action grammar rule with the highest probability is selected and used to explain the input symbols generated by the primitive action detectors. Third, although it shares many properties with HMM, it allows to express more general task structures, e.g. counting models such as $a^n b^n$. Last but not least, because of its compact representation using linguistic constructs, it allows a wide range of users to define actions which does not require high level of technical skills. It is worth noting

that the term "context-free" in SCFG is used as a contrast to "context-sensitive", which is another type of grammars, instead of meaning that it lacks contextual knowledge.

An action is defined using terminals, non-terminals and rule probabilities. A terminal, conventionally written in lower case, is generated by a primitive action detector with an associated probability. It can be easily added by defining an additional event detector. A non-terminal, conventionally written in upper case, is an intermediate symbol that can be regarded as a higher-level description. Rule probability, similar to transition probability in HMM, is applied when the state is expanded.

In (Ryoo and Aggarwal, 2007), Ryoo defines a game activity representation using context-free grammars (CFG) which enables a system to recognize events and actively provide proper feedback to the human user when the user makes unexpected actions. In (Ivanov and Bobick, 2000), Ivanov *et al.* define a task using SCFG, a probabilistic version of CFG, to recognize more complicated set of actions, e.g. music conducting gestures. They use HMM to detect primitive actions such as lifting a hand. In (Ota, Yamamoto, Nishimoto, and Sagayama, 2008), Ota *et al.* use SCFGs to describe the structures of Kanji using few stroke shapes and relative position labels. In (Kitani, Yoichi, and Sugimoto, 2008), Kitani *et al.* use SCFG to model complex employee-customer transaction activities in a convenience store, whereas Moore *et al.* use SCFG to model the Black Jack card game (Moore and Essa, 2002). In (Lee and Demiris, 2011), a robot imitates human demonstrations of organizing objects with a box using SCFG as task templates and associating each object with a corresponding grammar, where primitive actions such as *hand approaching an object* are defined using HMM.

Rizzolatti *et al.* (Rizzolatti and Arbib, 1998) propose that the action ob-

servation/execution matching system provides a necessary bridge from 'doing' to 'communicating', as the link between actor and observer becomes a link between the sender and the receiver of each message. They posit that action-recognition mechanism has been the basis for language development. An in-depth analysis of the structure of actions and its relation to human actions are presented in (Pastra and Aloimonos, 2012), while Aloimonos *et al.* (Aloimonos, Guerra-Filho, and Ogale, 2009) give a detailed background review on relationship between human actions and formal languages, as well as applications to health, artificial intelligence and cognitive systems. For other interesting areas that utilize CFGs as the underlying framework, e.g. computational biology and speech recognition, please refer to (Higuera, 2005).

2.3.1. Task Representation through SCFG

In this thesis, a task is represented as a grammar, which consists of terminal symbols, non-terminal symbols, production rules and a start symbol. The terminal symbols correspond to primitive actions, a basic set of human actions, which are the output of human action detectors. In this way, input signals from the sensory input are converted into a stream of terminal symbols. Non-terminal symbols represent abstract symbols that consists of one or more terminal and/or non-terminal symbols. As a convention, terminals are represented with lower-case letters and non-terminals are represented with upper-case letters. The production rules express how a non-terminal can be expanded, where in SCFG, every production rule is associated with a rule probability value. The start symbol is a non-terminal symbol that defines the starting state of the parser. Throughout the remaining of the thesis, the following notations will be used:

Σ: A set of terminal symbols

N: A set of non-terminal symbols

R: A set of rule productions

S: The start symbol

Given any $X \in N$, a production rule $r \in R$ is in the form of $X \rightarrow \lambda_1|\lambda_2|...|\lambda_n$, where $\forall k\ \lambda_k \in (\Sigma \cup N)^*$ and $\sum_{j=1}^{n} P(X \rightarrow \lambda_j) = 1$.

To illustrate how we can represent a task using SCFG, suppose the following scenario. Assume that a robot is given a task to enter one or more rooms, meet a person, and come back to the initial location. Assume that there can be more than one door to reach the target room. Consider the following simple grammar:

S	$\rightarrow$	ROOM	[1.0]
ROOM	$\rightarrow$	enter ROOM exit	[0.7]
ROOM	$\rightarrow$	enter meet exit	[0.3]

Figure 2.1.: A simple grammar that represents a task where a robot enters one or more rooms until "meet" event is happened, and returns to the initial location.

Figure 2.1 shows an example where "enter", "meet", and "exit" are terminals, and "S", "ROOM" are non-terminals. This grammar representation enforces a robot to open and close the same number of doors when going in and out of the room. Depending on the required resolution of task representation, it is always possible to further break down "enter", "meet", and "exit" actions into several sub-actions. The issue of defining the scope of a symbol will be discussed in Chapter 5. Since the recursion rule (ROOM $\rightarrow$ enter ROOM exit [0.7]) has a relatively higher probability, this grammar prefers an input sequence with longer length. It is also worth noting that

this type of task cannot be represented using HMM. For example, HMM cannot represent tasks of the form $a^n b c^n$, e.g. the task defined in Figure 2.1, where a, b and c are primitive actions and n is the number of execution.

2.3.2. Task Recognition through Probabilistic Parsing

Given a task representation G in an SCFG form, it is possible to parse the input symbols and compute the likelihood $P(input|G)$ as well as most likely parse tree. The parsed result provides two important information: task classification and observation error correction. Among multiple grammars, or tasks, it is possible compute choose the grammar G^* that best explains the observation, i.e.

$$G^* = \arg\max_j P(input|G_j) \tag{2.1}$$

In this thesis, the input terminal symbols are parsed using a probabilistic version of the Earley parser, which not only considers the rule probability but also the terminal symbol probability (Ivanov and Bobick, 2000). As in the Earley parsing algorithm, the notion of a *State* is expressed in the following way.

$$i : X_k \rightarrow \lambda . Y \mu \tag{2.2}$$

where '.' is the marker of the current position in the corresponding production., λ and μ are strings of mixed terminal and non-terminal symbols with arbitrary length. In the above notation, i is the current marker position in the input and the non-terminal symbol X is expanded into $\lambda Y \mu$, starting at the position k in the input, generating some substring starting at position k. Given an input $w_1 w_2 ... w_k ... w_m$, where w_1 is the first terminal

symbol and w_m is the last terminal symbol of μ, a non-terminal X in the above case is said to *dominate* the substring $w_k...w_m$. For each position, the parser generates a set of possible states where each state "explains" the strings which it dominates. The parsing is done by iteratively applying the following transitions.

Scanning

During the scanning step, the parser reads the next input symbol and verifies it against all pending hypotheses states. Assuming that a is a terminal symbol of the current input, a new state is added for every rule expecting the terminal symbol a.

$$i : X_k \to \lambda.a\mu \quad [\alpha, \gamma] \Rightarrow i+1 : X_k \to \lambda a.\mu \quad [\alpha', \gamma']$$

$$\begin{aligned} \alpha' &= \alpha(i : X_k \to \lambda.a\mu)P(a) \\ \gamma' &= \gamma(i : X_k \to \lambda.a\mu)P(a) \end{aligned} \tag{2.3}$$

$$\forall a, s.t. P(a) > 0$$

where α and γ are forward and inner probabilities, similar to the notion used in Hidden Markov models, and the expression $A \Rightarrow B$ is used in the context of state generation to read "A generates B".

Note the increase in i index. This signifies the fact that scanning step inserts the states into a new state set for the next iteration of the algorithm.

$P(a)$ accounts for uncertainty in the input, i.e. the probability value of the low-level detector. To handle *substitution error*, which occurs when an incorrect terminal replaces the correct one in the input stream, input to the parser is an array of all possible terminal probabilities.

Completion

Completion uses the results of scanning to advance positions in the parse tree. In its simplified probabilistic form, completion step generates following states:

$$\begin{cases} j : X_k \to \lambda . Y \mu \quad [\alpha, \gamma] \\ i : Y_j \to \nu . \quad [\alpha'', \gamma''] \end{cases} \Rightarrow i : X_k \to \lambda Y . \mu \quad [\alpha', \gamma'] \tag{2.4}$$

where

$$\alpha' = \sum_{\forall \lambda, \mu} \alpha(i : X_k \to \lambda . Y \mu) \gamma''(i : Y_j \to \nu .)$$
$$\gamma' = \sum_{\forall \lambda, \mu} \gamma(i : X_k \to \lambda . Y \mu) \gamma''(i : Y_j \to \nu .)$$

Due to possible recursions, recursive correction needs to be applied to probability computations. This topic is out of the scope of this paper, and we direct you to (Ivanov and Bobick, 2000) for further reading.

Prediction

Prediction step is used to hypothesize the possible continuation of the input based on the current position in the parse tree:

$$\begin{cases} i : X_k \to \lambda . Y \mu \quad [\alpha, \gamma] \\ Y \to \nu \end{cases} \Rightarrow i : Y_i \to . \nu \quad [\alpha', \gamma'] \tag{2.5}$$

where

$$\alpha' = \sum_{\forall \lambda, \mu} \alpha(i : X_k \to \lambda . Y \mu) P(Y \to \mu)$$
$$\gamma' = P(Y \to \nu)$$

The prediction step introduces the rule probabilities $P(Y \to \nu)$ associated

with the productions $Y \rightarrow \nu$ into the parsing process. Similar to *Completion* step, recursive correction is required due to possible recursions.

As discussed in (Stolcke, 1995), the time complexity of Earley's parser is $O(l^3)$, where l is the length of symbols. It decreases to $O(l^2)$ if a grammar is unambiguous, i.e. the number of distinct derivation trees of a sentence is 1.

2.4. Task Structure Learning

The previous sections only considered cases where the grammars are known in prior for the recognition of human behaviors. However, it is not sufficient for a robot to automatically learn a novel task without the capability to discover the hidden task structure or at least partial task structures from human demonstrations. Hence, as opposed to manually defining a grammar to represent a task, there are also several approaches aiming at learning (or *inducing*) grammars from observation data.

In early work, Nevill-Manning *et al.* (Nevill-Manning and Witten, 1997) presented the SEQUITUR algorithm which can discover hierarchical structures among symbols. Solan *et al.* (Solan, Horn, Ruppin, and Edelman, 2005) presented the ADIOS algorithm which induces CFGs and context-sensitive grammars as well, with some restrictions (e.g. no recursions) using graphical representations. Stolcke and Omohundro (Stolcke and Omohundro, 1994) presented an SCFG induction technique, which more recently has been extended by Kitani *et al.* (Kitani, Yoichi, and Sugimoto, 2008) to remove task-irrelevant noisy symbols to cope with more realistic environments. In (Ogale, Karapurkar, and Aloimonos, 2007), Ogale *et al.* construct an SCFG grammar based on the frequency of human pose pairs, i.e. bi-

grams, considering slightly varying viewpoints. However, it does not have a generalization step which differs from our approach.

In (Stolcke and Omohundro, 1994), Stolcke and Omohundro proposed a technique on merging states which generalizes SCFG rules to deal with unforeseen input with arbitrary lengths, e.g. symbols generated using recursive representations. They introduce two operators, chunking and merging, which convert an initial naive grammar to a more general one. More recently, Kitani *et al.* (Kitani, Yoichi, and Sugimoto, 2008) presented a framework of discovering human activities from video sequences using an SCFG induction technique based on (Stolcke and Omohundro, 1994). By assuming that the noise symbols are not part of the task representation, they try excluding some symbols from input stream until a grammar with strong regularity is found based on minimum description length (MDL) principle. Our approach is different from these, however, in that it takes into account the uncertainty (or probability) values of input symbols and explicitly searches for a multiple set of regularities in input symbols. This allows our method to learn a grammar that reflects the noise term included in the observation, as well as structure components that form as part of the whole structure.

In the human-robot interaction domain, Nicolescu and Mataric (Nicolescu and Mataric, 2003) presented a framework which generalizes graph-based task representations by merging nodes to induce a graph with the longest common sequences. After learning, they allow their system to interactively modify the task representation from human vocal commands. The notion of nodes in their work corresponds to that of our non-terminal symbols which are essentially state representations. However, as their framework is inherently based on directional acyclic graphs, it cannot induce a representation containing recursive actions, which is often useful to describe hierarchical

human tasks.

2.5. Task Structure for Attention Control

Active localization and detection of task-relevant activities play an important role in many robotic systems where the computational resources are often limited. In situations where the goal is to locate and recognize a set of critical actions from the limited range of camera view and limited computational resources, it is critical for a robot to keep actively search for the best information source. Since this problem can be formulated as an attention problem, we present a review in this field.

The human ability to actively allocate fixation points in high resolution imagery of fovea presents several advantages such as invariance to large translations and reduction of the sensory input size while preserving the ability to perceive fine details. Additionally, selective attention may allow to reduce the representation costs by removing irrelevant input signals (Larochelle and Hinton, 2010; Ballard, 1991; Suzuki and Floreano, 2008). These advantages have been adopted by the active vision paradigm (Aloimonos, Weiss, and Bandyopadhyay, 1988; Bajcsy, 1988; Borji and Itti, 2013) and in several applications such as object detection (Vijayanarasimhan, Jain, and Grauman, 2010; Vogel and Freitas, 2008; Croon and Postma, 2007), object recognition (Denzler and Brown, 2002; Larochelle and Hinton, 2010; Paletta, Fritz, and Seifert, 2005), monitoring during action execution (Sridharan, Wyatt, and Dearden, 2010), and tracking (Sommerlade and Reid, 2008; Gould, Arfvidsson, Kaehler, Sapp, Messner, Bradski, Baumstarck, Chung, and Ng, 2007; Ognibene, Pezzulo, and Baldassarre, 2010). Due to this reason, active vision systems have been gaining more interest

in the robotics and vision communities as their performances are comparable or even surpass conventional passive vision systems (Larochelle and Hinton, 2010; Andreopoulos and Tsotsos, 2013; Denzler and Brown, 2002). They have also been applied with success in dynamic context like tracking and event recognition (Yu, Fermuller, Teo, Yang, and Aloimonos, 2011; Ognibene and Demiris, 2013; Oliver and Horvitz, 2005).

Several attention systems have been extensively studied to design bottom-up, e.g. (Itti, Koch, and Niebur, 1998; Denzler, Zobel, and Niemann, 2003) and top-down attention systems. Since we are interested in exploiting the high-level task knowledge to control the attention, we will focus on top-down attention systems that deal with vision problems. For bottom-up attention systems, Itti *et al.* proposed a bio-inspired model (Itti, Koch, and Niebur, 1998) and with an information-theoretic principle (Itti and Baldi, 2005), where as (Denzler, Zobel, and Niemann, 2003) proposed an automatic camera zoom parameter selection for object tracking based on the uncertainty outputs of Kalman filters.

Prior works that use information-theoretic principles for detection and tracking include (Sommerlade and Reid, 2008; Sommerlade and Reid, 2010; Oliver and Horvitz, 2005; Yu, Fermuller, Teo, Yang, and Aloimonos, 2011). Although (Sommerlade and Reid, 2008; Sommerlade and Reid, 2010) use mutual information maximization for multi-target detection and tracking, the decision is made from object motion information, which does not rely on a generative activity model. Although (Oliver and Horvitz, 2005) represents human actions using layered hidden Markov models, they model only unstructured actions such as phone conversation or face-to-face conversation.

Oliver and Horvitz (Oliver and Horvitz, 2005) propose an activity recognition system using Layered Hidden Markov Models (LHMM) which decides

when to extract and process multi-modal sensory information to maximize the expected value of information. It selects which low-level features to extract from the camera, e.g. motion density, face and skin color density, to improve the estimation of the higher-level LHMM models. While it shares a similar theoretical basis, our approach focuses on recognizing temporally extended activities, whereas (Oliver and Horvitz, 2005) deals with more instantaneous actions, e.g. user present/absent, phone conversation, etc. Furthermore, we consider the recognition of concurrently occurring multiple activities.

Yu *et al.* (Yu, Fermuller, Teo, Yang, and Aloimonos, 2011) propose a system that controls attention to actively recognize the category of a scene or an activity by sequentially finding the most relevant objects and movements. Their method uses maximization of mutual information to select the next area to analyze and what object to look for. Their approach aims to recognize a single short-term activity, like painting or cutting, finding the related objects and movements in predicted positions.

There are few other works in the literature that use higher level temporal knowledge to optimize the use of computational resources for the perception of an event. Chen *et al.* (Chen, Bilgic, Getoor, and Jacobs, 2011) present an algorithm to decide which frames of a video should be analyzed with expensive algorithms to detect motion events and faces in videos. These expensive observations are initially made at low fixed frame rate, where additional information is provided by fast detection algorithms. While their approach is applied at the end of the observation to select which frame to analyze, our method selects which part in a frame to observe at each time step, which is an online system.

Ognibene *et al.* (Ognibene and Demiris, 2013) propose an approach to

control online the cameras of a robot to recognize simple events, like reaching actions of a human partner toward targets of uncertain position. They model the set of possible events as a mixture of Kalman Filters and use the maximum information gain to select the camera target between the human partner or the action target candidates. However, this method differ from our approach since they rely only on object position information, from which both motion and event uncertainties are computed.

2.6. Summary

In this chapter, related works in the Learning from Demonstrations (LfD) paradigm, task representation and recognition, task learning and anticipation have been reviewed. With the currently available frameworks, however, it is not possible to realize a robot that learns and represent temporally structured tasks from human demonstrations efficiently. While positioning itself within the LfD paradigm, this thesis presents a novel and compact task representation and recognition framework that are well fit for robot imitations, as well as automatically learning tasks from human users.

The following chapter starts by discussing efficient representation of human tasks that could be used as task templates and able to deal with observation uncertainties.

3. Syntactic Approaches to Task Representation and Recognition

3.1. Introduction

In this chapter, we investigate how the knowledge of tasks can be exploited to recognize human behaviors with the purpose of better imitation in Learning from Demonstration (LfD) paradigm through SCFG. This is done through exploiting the semantic constraints of a task, which results in recognizing the human's intention correctly and imitating the correct actions. It results in a more efficient and natural interaction between a human user and a robot by minimizing the need to correct the errors made by the robot while executing actions. It will be also shown how the probabilistic parsing is used to facilitate this process. We demonstrate our implementation on a real-world scenario using a humanoid robot and report implementation issues we had.

3.2. Experiments

This experiment investigates how a robot can recognize human demonstrations and execute demonstrated actions correctly while inferring human's intention in case the observation was not fully reliable. For this purpose,

a set of hierarchical, problem-independent task templates are given to the robot in the form of SCFG, which will be later instantiated to problem-dependent task representations by users that suit their needs, as shown in Figure 3.1. This process of contextualization is done through SCFG parsing.

The approach presented here shares the same concept as done in (Demiris and Khadhouri, 2006; Ognibene, Wu, Lee, and Demiris, 2013) where the authors designed a set of primitive actions which are then used as basic building blocks, i.e. basic vocabularies, of higher-level tasks. In this experiment, however, more complex topics such as the probabilistic representation of actions as well as recursive representations of the task structure are also studied.

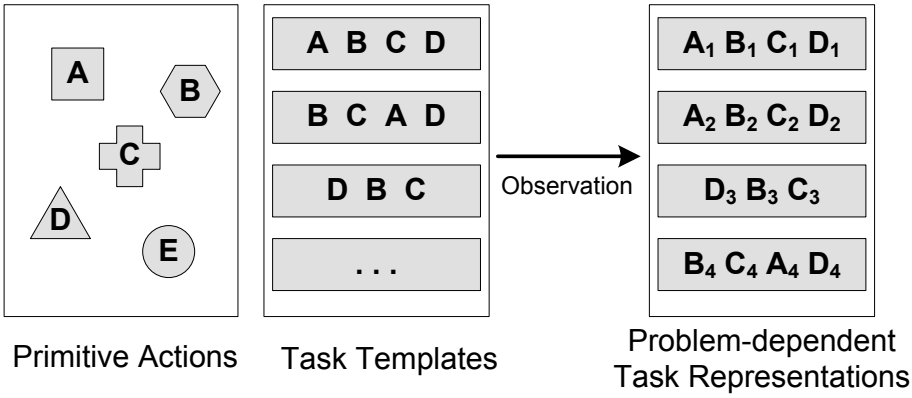


Figure 3.1.: Building problem-dependent task representations from a pool of generic task templates. Primitive action detectors are trained offline to convert the input signals into the time-series terminal symbols for the SCFG parser. By observing a human demonstration and parsing the observation, the system classifies the demonstration into a corresponding task and assigns the task property, e.g. 1, 2, ... 4.

We use the hand as a reference cue that describes the observation. Flanagan and Johansson (Flanagan and Johansson, 2003) demonstrated in their experiments that when people watch a series of block-moving tasks, they

tend to map the visual representation of the observed action onto a motor representation of the same action, instead of a purely visual analysis of the elements independent from actuators. In both our and Flanagan's cases, hand is equivalent to the actuator which forms the basis of the visual representation of objects.

3.2.1. Experiment Design

In this experiment, a human participant teaches a robot how each object should be organized using a box among three types of organization methods. The robot observes a human organizing an object using a box, finds a task template that best matches with the observed demonstration, and instantiate the task template by associating it with the object used in the demonstration. Depending on the object type, each object should be treated differently: while some objects can be dropped into the box, fragile objects need to be placed safely inside the box, whereas over-sized objects should be put next to the box. We call these tasks respectively as DROP (drop an object into the box), PLACE (place an object inside the box), and NEXTBOX (place an object next to the box). The human participant decides what kind of object to be used for each organization task. After demonstrations are finished, the robot confirms by executing the parsed primitive actions of each task with the associated object. This process is summarized in Figure 3.2. [1]

We conduct our experiments with 10 participants repeating 10 demonstrations each, where each demonstration consists of three different tasks. A participant sits on a chair approximately 1.2m distant from the robot where a table is placed in the middle, although the participant is allowed

[1] This experiment appears in (Lee and Demiris, 2011).

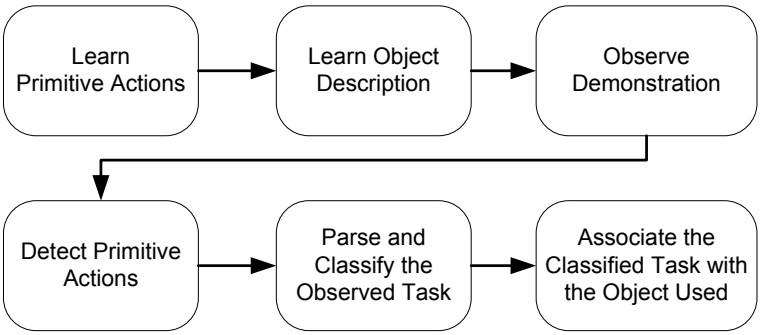


Figure 3.2.: The procedure of the experiment.

to sit a little bit closer or farther from the robot if it makes them feel more comfortable. The participant starts the experiment by arbitrarily selecting an object among three different cotton dolls, a ceramic doll, two types of fruits and a water bottle, and performs a task using the chosen object. The task order, as well as the choice of an object for each task is fully up to the demonstrator's will. There is also no restriction on the demonstrator's performing speed and movement trajectories as long as they think it is natural.

After finishing the demonstrations, the participant places 3 objects that were used in the demonstration in front of the robot, where the robot then performs the parsed primitive actions with the associated object by pointing to each object and showing the corresponding task using gestures. The reason it shows gestures instead of actually manipulating the objects is because of the grasping limitation issues with the hand of our iCub. The quantitative analysis of task accuracy as well as qualitative analysis are given in Section 3.2.2.

Although providing generic task templates in prior might seem heuristic to some extent, we posit that these task-level representations are crucial for natural and efficient interactions between humans and robots instead of

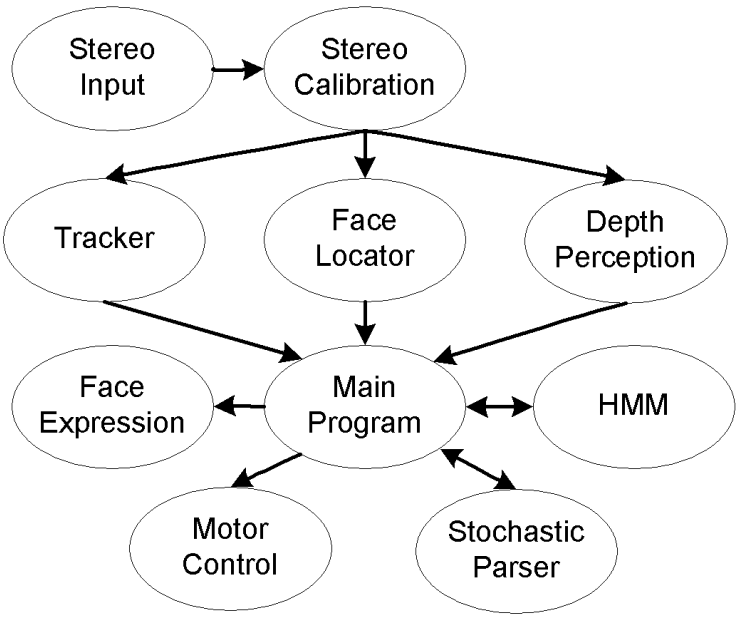


Figure 3.3.: Software modules used in the experiment.

learning from the scratch.

Task Representation and Parsing

The tasks are defined as shown in Figure 3.4. The non-terminal symbols that self-represents terminal symbols such as AOBJ and CONTACT are added to handle repetitive symbols and erroneous symbols. In practice, there are two important problems to be considered while parsing noisy data:

1) Symbol substitution problem: When an incorrect terminal symbol is detected in the place of the correct one due to observation noise.

2) Symbol insertion problem: When unexpected symbols are inserted which lengthens the overall observation. To handle this type of problem, we need to disregard them if their appearance in the input stream for some derivation is found to be incompatible. At the same time, we need to preserve the symbol in case it could satisfy other possible derivations.

An effective solution is to introduce the "NOISE" symbol. It can be thought of as a wildcard which can accept any symbol which enables the parser to allow some out-of-context, unexpected input symbols. It gives a "tolerance" to observation errors and it is typically set to expand with a low probability. If the primitive action detectors are unreliable, the overall parsed likelihood will be low, instead of simply rejecting the input. The concept of the NOISE symbol is similar to the notion of measurement noise covariance in Kalman filter, which depends on the reliability of the sensor. In our case, the probability of entering NOISE rule is set to 0.1 based on heuristics.

An SCFG parser receives input a sequence of N dimensional vectors where N is the number of terminals, i.e. the number of primitive action detectors. It then parses them to find the most likely parse tree that best explains the observation and outputs the overall likelihood of every grammar.

Visual Tracking

Based on the illustrated scenario, we implement the system as described in Figure 3.2. It first automatically learns the demonstrator's skin color histogram by extracting a patch from the detected demonstrator's face using boosted cascade classifiers (Viola and Jones, 2001) and uses it to track the demonstrator's hand. It subsequently learns the color histogram of the object chosen by the demonstrator.

We use hand and object trackers based on the CamShift tracking algorithm implemented in OpenCV library (Bradski, 2000), as they are fast enough to run in real time and robust to noise if the tracking object colors are distinctive with the following parameters: The number of histogram bins=32, minimum color saturation threshold=30, brightness acceptance

G_{DROP}:

S	→	AOBJ CONTACT ABOX LOBJ OGONE		[1.0]

G_{PLACE}:

S	→	AOBJ CONTACT ABOX OGONE HGONE		[1.0]

$G_{NEXTBOX}$:

S	→	AOBJ CONTACT ABOX LOBJ		[1.0]

AOBJ	→	AOBJ	aobj	[0.5]
	|	aobj		[0.4]
	|	NOISE	aobj	[0.1]
ABOX	→	ABOX	abox	[0.5]
	|	abox		[0.4]
	|	NOISE	abox	[0.1]
CONTACT	→	CONTACT	contact	[0.5]
	|	contact		[0.4]
	|	NOISE	contact	[0.1]
LOBJ	→	LOBJ	lobj	[0.5]
	|	lobj		[0.4]
	|	NOISE	lobj	[0.1]
OGONE	→	OGONE	ogone	[0.5]
	|	ogone		[0.4]
	|	NOISE	ogone	[0.1]
HGONE	→	HGONE	hgone	[0.5]
	|	hgone		[0.4]
	|	NOISE	hgone	[0.1]

Naming conventions:
A: approach, L: leave, OBJ: object, BOX: box
HGONE: hand visibility, OGONE: object visibility
CONTACT: hand in contact with an object
NOISE: (See Section 3.2.1)

Figure 3.4.: Task templates used in the experiment. Three different types of tasks are shown in the top part (G_{DROP}, G_{PLACE}, $G_{NEXTBOX}$), followed by non-terminal symbols commonly shared across three grammars.

range=20-240. Hand color histograms are learned from the face patch of the demonstrator in the beginning, and used throughout the experiment until all tasks have been performed. Object patches are obtained when a participant holds an object close enough to the robot, where its distance is computed from the depth perception using both cameras of iCub. The task is performed after the object has been learned.

The tracker is initialized using the following procedure:

1) Robot locates the face. Demonstrator holds an object close to the robot.

2) Obtain the distance of the face d_{face} from the depth image. Binary-threshold the depth image with threshold $d_{face}/2$. The result is called mask image.

3) Apply a mean-shift tracker on the mask image and wait until its center point P_{center} gets stabilized. Extract a small patch from the color input image around P_{center} and compute its color histogram. Example results can be seen in Figure 3.5.

4) Subtract the previously learned skin color histogram from the object color histogram, bin by bin, to filter out finger colors for better tracking performance. An example of the final result can be seen in Figure 3.6.

5) Robot nods to notify that it is now ready to track. Demonstrator places the object on the table and initial search window is set to the table area. When both hand and object are located, robot starts tracking them.

The method we used allows the system to learn an object in a natural way from humans with high success rate. It worked as expected on most of trials although there were occasionally flickering noise on the border area. In our experiments, wrong object patches were learned only 4 times out of 100 trials. An example object segmentation can be seen in Figure 3.5. We



Figure 3.5.: Examples of object segmentation. Images are acquired from the both eyes of iCub, from which depth map is computed and blobs closer than a threshold are segmented.

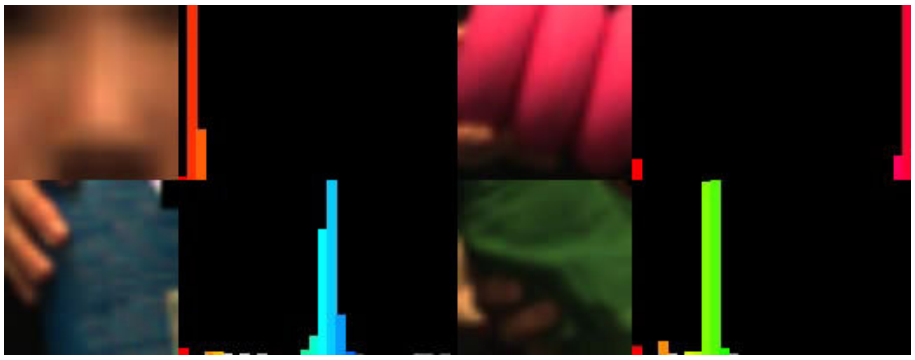


Figure 3.6.: Extracted patches and their color histograms. In histogram images, x-axis represents the color bin and y-axis represents the frequency. Finger colors in the patch are suppressed for better tracking performance.

average the positions of each tracker every 3 frames and use them as input to the primitive action detectors to increase the tracker stability.

Primitive Action Detectors

To recognize these tasks, it is a natural requirement for a system to be able to recognize the meaningful primitive actions such as a) hand approach or leave away from an object, b) grasp or release an object, c) move an object closer to the box. Before the demonstration, a set of generic-purpose primitive action detectors are trained offline. The choice of action learning technique is up to the system designer's decision, e.g. Hidden Markov Models (HMM) or Conditional Random Fields (CRF) (Ahad, Tan, Kim, and Ishikawa, 2008), where in our experiment we employ HMM for the recognition of dynamic primitive actions. We then define task templates in the form of SCFG using these primitive actions as terminal symbols. The output of the primitive action detectors become the input to the SCFG parser.

Primitive action detectors compute the probability of certain types of events being occurred from the input data in parallel for every time step. Examples include low-level motions such as `hand approaching an object` and object states such as `object observable`. As long as they provide the probability or confidence values between 0 and 1, any primitive action detectors can be used, e.g. aural or tactile event detectors. The output values represent the probability of terminal symbols. In our case, we use 7 primitive action detectors in total, as described below. We denote H for hand, O for object, and B for box.

1) 'H approaching O', 'H leaving away from O', 'O approaching B', 'O leaving away from B': They represent the relationships between two entities. The system learned two general types of HMM, 'Approaching' and 'Moving

away from' offline using 20 tracked video samples. The input to each HMM is the sign change of distance between two entities, i.e. {-,+,0}.

2) 'Object visibility' and 'Hand visibility': These two symbols represent the observability of objects. Probabilities are obtained by computing the Bhattacharyya distance between the histogram of the current object tracking window and its previously learned histogram.

$$dist(H_1, H_2) = \sqrt{1 - \frac{1}{k}\sum_i \sqrt{H_1(i)H_2(i)}}$$

where $H(i)$ = the i-th bin value of a normalized histogram H, and

$$k = \sqrt{\sum_i H_1(i) \sum_i H_2(i)}$$

A color bin size of 32 is used for the experiment. The above function outputs the histogram distance between 0 and 1, where 0 means two histograms are identical. Ideally, if an object is placed in a box, its visibility should reach 1.

3) 'In contact with an object': This detector is a Gaussian function with parameters learned from 50 samples of distances between hand and object center positions while holding an object.

3.2.2. Findings

A total of 100 sets of experiments were performed, excluding 6 sets that were not usable due to recording problems. Typical single task demonstration spans between 2 and 6 seconds excluding object learning time. In some extreme cases, actions were extremely fast (less than 1 second) or slow (more than 20 seconds). In this experiment, only the performance of recognizing actions is evaluated, not object recognition, as the latter belongs to another

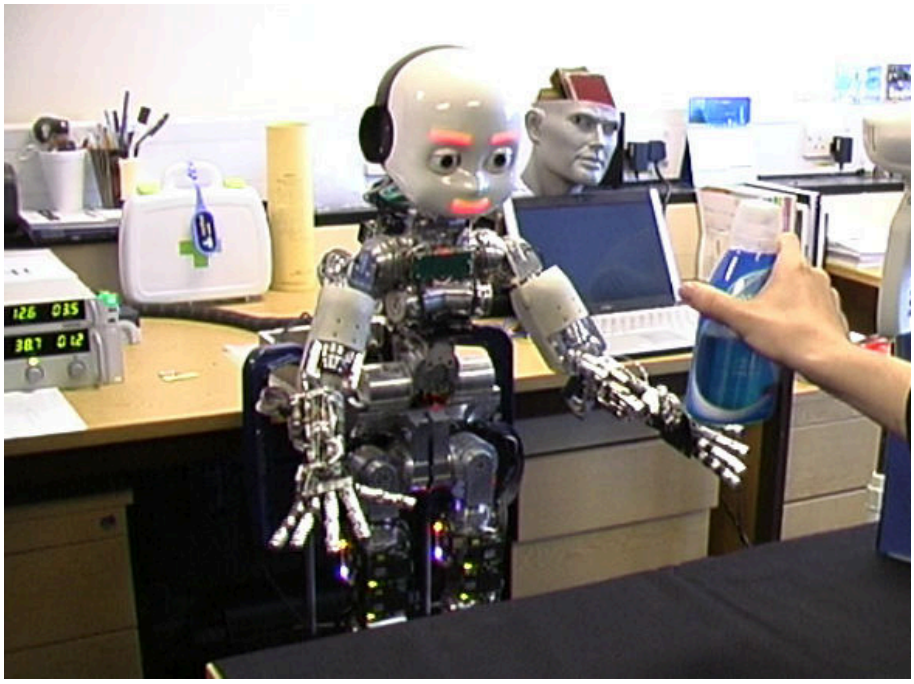


Figure 3.7.: iCub observing the object organization task demonstrated by a human participant.

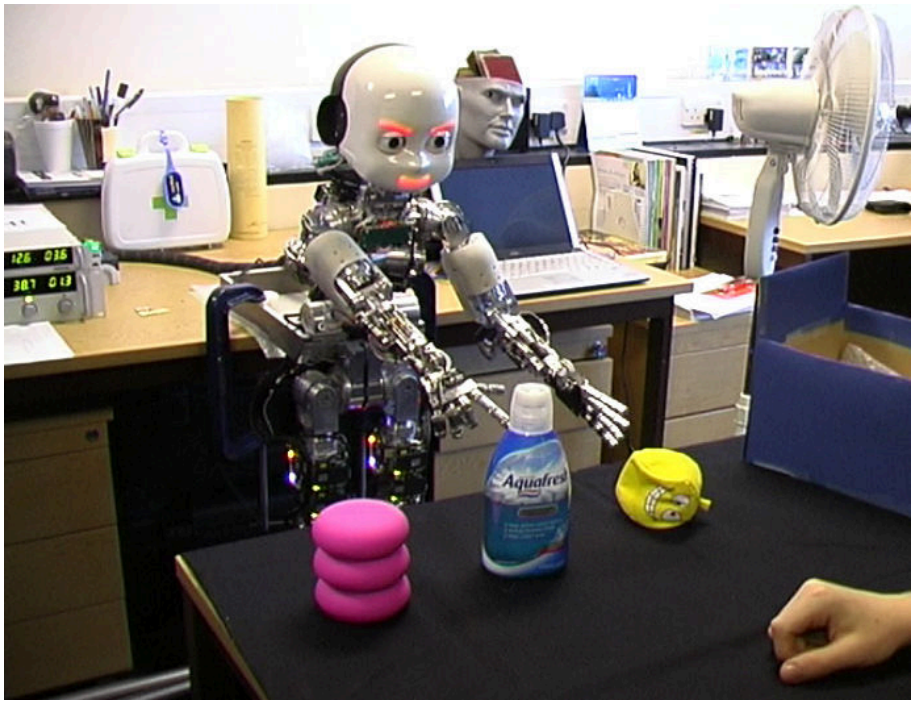


Figure 3.8.: iCub performing task by executing parsed primitive actions.

problem domain.

Figure 3.9 shows an example output of primitive action detectors and the parsed result obtained by the stochastic parser. The terminal symbols in the last line denotes the most probable terminal path reached based on the overall observation. Figures 3.10 and 3.11 shows the raw scores and confusion matrix, respectively.

It is worth noting that "aobj" (approach object) symbol in Figure 3.9 has a low likelihood on time steps 2 and 3 (0.0336 and 0.0512, respectively), which is supposed to have a higher likelihood as the "DROP" task expects to observe only "aobj" symbols until the object is grasped. However, after the demonstration is recognized as a "DROP" task, these ambiguous symbols are parsed correctly as "aobj" to be consistent with the task representation.

Figure 3.10 shows the actual number of trials and errors made in this experiment. "Gt" denotes the ground truth while "Ob" denotes the observed result. "X" denotes the case where the algorithm fails to find the answer due to extremely low probabilities. Although rare, it occurs if there are too many symbols that are inconsistent with all of the defined rules.

Figure 3.11 shows the confusion matrix of the overall result. The accuracy of recognizing the NEXTBOX tasks is high because it is relatively easier to recognize this task due to its simple structure. (Defined in Figure 3.4) The PLACE tasks were recognized as DROP in more than 20% of the trials. This is mainly due to the ambiguity between two similar grammars.

If the demonstration is done too slowly, the tracker often suffers from "jitter" effect which increases the error on the output. This problem could be alleviated by applying Kalman filter on the tracker although we have not used it in this work. As can be seen on time steps 2 and 3 in Figure 3.9, even when the hand was approaching the object, "approaching" action was

time	abox	lbox	aobj	lobj	contact	ogone	hgone
1	0.1174	0.1426	0.6868	0.0532	0.0000	0.1800	0.1985
2	0.5284	0.0136	0.0336	0.4245	0.0000	0.3826	0.1726
3	0.4796	0.0216	0.0512	0.4476	0.0000	0.3627	0.2095
4	0.2098	0.0640	0.6849	0.0413	0.0000	0.3103	0.2053
5	0.1590	0.0681	0.7359	0.0370	0.0000	0.3186	0.3366
6	0.1598	0.0654	0.7477	0.0270	0.0001	0.1427	0.5125
7	0.1208	0.0930	0.7614	0.0248	0.0013	0.2728	0.5846
8	0.3048	0.0277	0.6464	0.0210	1.0000	0.2159	0.6022
9	0.3261	0.0254	0.6296	0.0189	1.0000	0.1977	0.6196
10	0.2905	0.2511	0.1193	0.3392	0.0000	0.8438	0.2689
11	0.3092	0.2697	0.1366	0.2846	0.0000	0.8446	0.2708
12	0.4722	0.4753	0.0328	0.0197	0.0000	0.8549	0.2335

(The actions with the highest likelihood are underlined.)

>> Task recognized as DROP.
>> Parsed actions assuming it was a DROP demonstration:
aobj aobj aobj aobj aobj aobj aobj contact abox lobj ogone ogone
(The corrected actions after parsing are underlined.)

Naming conventions

abox, aobj:	Approach box, Approach object
lbox, lobj:	Leave box, Leave object
hgone:	Hand visibility (1 invisible, 0: visible)
ogone:	Object visibility (1 invisible, 0: visible)
contact:	Hand in contact with an object

Figure 3.9.: Sample terminal symbols generated by primitive action detectors and the parsed result. The action symbols with the highest likelihood are underlined. The likelihood values of aobj in time steps 2 and 3 are unexpected, which are corrected after concluding that the demonstration was a DROP task. For naming conventions, please refer to Figure 3.4.

Gt \ Ob	**N**	**D**	**P**	**X**	**Sum**
N	85	7	0	2	94
D	8	76	7	3	94
P	7	22	60	5	94
Sum	100	105	67	10	282

Figure 3.10.: N:Nextbox, D:Drop, P:Place, X:Recognition Failure

Gt \ Ob	**N**	**D**	**P**	**X**
N	0.90	0.07	0.00	0.02
D	0.09	0.81	0.07	0.03
P	0.07	0.23	0.64	0.05

Figure 3.11.: Confusion Matrix

detected with low likelihood and "leaving" as high likelihood.

3.3. Summary

This chapter outlines a robot imitation method using SCFG as task templates. Primitive action detectors are trained and used to generate probabilistic terminal symbols which are used to parse higher-level tasks. The task properties, selected by a human user, are assigned to a task template, making it a problem-dependent task representation. The robot performs the correct task when a similar object is found. Two levels of representations (task-level and action-level representations) are used to fulfill this purpose. The task-level representation is used as a semantic constraint that enforces the ambiguous observed actions to fit into its context. Any low-level detectors can be designed that best suit the situation which makes

this approach scalable. Another advantage of this approach is that human-readable grammar-like representations are intuitive and easy to understand, which will allow a wider range of users to take full advantage.

In this experiment, the grammars were hand-crafted, which is common in many domains as discussed in Section 2.3. However, it is possible to learn the structure and probabilities of rules although it is commonly regarded as intractable (Higuera, 2005). In the following chapter, Chapter 4, we investigate on the problem of learning grammars from human demonstrations. Furthermore, we investigate the problem of learning primitive action detectors in an unsupervised manner in Chapter 5.

One limitation in this experiment is that it is necessary to know when to start and stop observing. It is possible to work-around this problem by adding vocal commands or specific gestures made by the demonstrator, but they are essentially still equivalent to manual manipulation. However, this topic is out of the scope of this thesis.

On the execution part, it is possible for a robot to run each primitive action with the same timing as it had learned from the demonstrator. It is worth noting that by recording the actual timing between actions, it is possible for the learner to execute the parsed action at the right timing. Although execution timing was not critical in our example, one could easily imagine other kinds of tasks where it is more important, e.g. playing musical instruments.

So far we have discussed on the use of task templates using SCFG to correctly understand what humans are doing, where the tasks were defined manually. To make it more useful in the human-robot interaction environments, it is desirable to make the system learn new tasks automatically from human users. In the following chapter, the mechanisms of learning

novel tasks from human users will be presented, along with several different experiment scenarios.

4. Learning Task Structures from Demonstrations

4.1. Introduction

Humans are capable of learning novel task representations despite noisy sensory input by making use of previously acquired contextual knowledge, since many human activities often share similar underlying structures. For example, when we observe a hand transferring an object to another place where a grasping action cannot be seen due to some occlusions, we can still infer that a grasping action occurred before the object was lifted.

Similarly, in the process of language acquisition, a child learns more complex concepts and represents them by using previously learned vocabularies. Analogously, the structure of a task can be represented using a formal grammar, where the symbols (or vocabularies) represent the basic action components, i.e. primitive actions. We are interested in learning reusable task components to better understand more complicated tasks that share the same task structures under noisy environments.

The learning of reusable task components is one of the crucial tools in LfD as it enables a robot to incrementally learn higher-level knowledge from human teachers using the previously learned knowledge. In this respect, we present a computational model of the structured human task learning

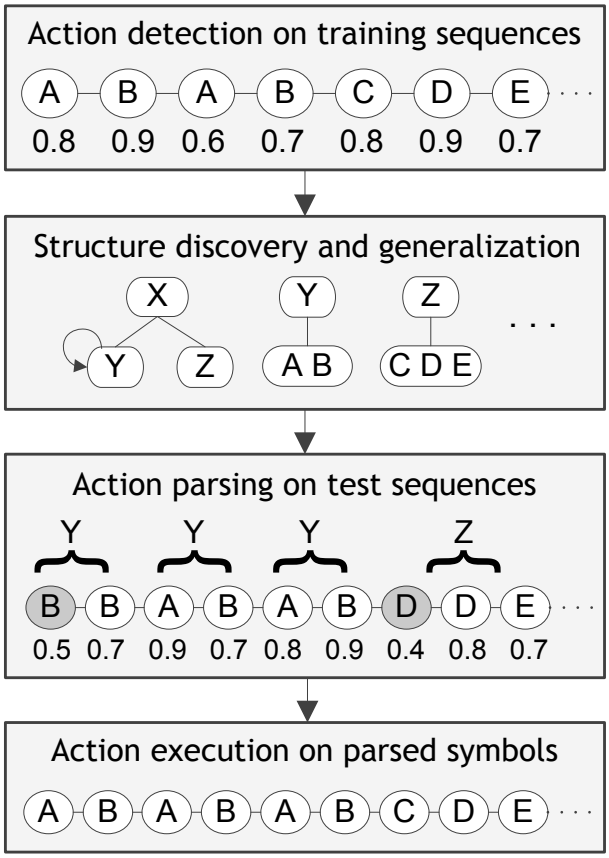


Figure 4.1.: Overview of the approach to task learning with an example scenario. The input training sequences are converted into streams of symbols with probability, respectively indicated by circles and numbers below, from which the original structure is uncovered using grammatical representations. The acquired knowledge is used to better recognize unforeseen, more complex tasks (test sequences) that share the same structure components.

using grammars from demonstrations which can be subsequently used as a prior knowledge to better recognize more complex tasks that share the same underlying components with ambiguity. We assume that 1) the system can detect meaningful primitive actions which are not necessarily noise-free, and 2) extensive complete data sets are not always available but numerous examples of smaller component elements could be found.

Compared to the conventional learning techniques, our method has two distinctive features: 1) Our method actively searches for frequently occurring sub-strings from the input stream that are likely to be meaningful to discover hierarchical structures of a task. 2) We take into account the uncertainty values of the input symbols computed by primitive action detectors. Figure 4.1 gives an overview of our approach with an example for illustrative purpose. This is inspired by Ivanov's work (Ivanov and Bobick, 2000) where they augmented the conventional SCFG "parser" by considering the uncertainty values of the input symbols. We augment the conventional SCFG "induction" technique by considering the uncertainty values of the input symbols.

4.2. The Discovery of Task Structures and Parameters

To induce a task grammar from input data, which is a sequence of time-series terminal symbols, first an initial naive grammar is built as the starting point by adding all input sequences to the start symbol S. Starting from the initial grammar, two kinds of operators, *Substitute* and *Merge*, are applied until the grammar is found. The quality of a grammar is measured by the Minimum Description Length (MDL) principle as used in (Langley and Stromsten,

2000; Kitani, Yoichi, and Sugimoto, 2008; Stolcke and Omohundro, 1994), which will be explained more in Section 4.2.3. In the context of robot learning of human tasks, the technique of merging repetitive symbols used in (Nevill-Manning and Witten, 2002) can be reinterpreted as a means of abstracting meaningful action symbols into hierarchical structures.

There are two operators that abstract and generalize the initial grammar. The *Substitute* operator builds hierarchy by replacing a partial sequence of symbols in the right-hand side of a rule with a new non-terminal symbol. The new rule is created such that a new non-terminal symbol expands to these symbols. The $Merge$ operator generalizes rules by replacing two symbols with the same symbol. $Merge(X, Y)$ *into* Z means all X and Y symbols in production rules are replaced with the symbol Z. As a result, it converts the grammar into the one that can generate (or accept) more symbols than its predecessor while reducing the total length of the grammar.

The challenging problem here is that there is no obvious way to efficiently choose which operator to apply. In case of HMMs, choosing the locally best choice (greedy strategy) generally leads to good results (Stolcke and Omohundro, 1994). However, it is no longer the case in SCFGs as *Substitute* operator often requires several following $Merge$ or *Substitute* operators to produce a better grammar. We use a beam-search method to limit the search space, which considers a number of relatively good grammars in parallel and stops if certain neighborhood of alternative models has been searched without producing further improvements. We also take advantage of the beam search strategy with depth 3, which is reported to find most of the important grammatical structures of SCFG (Stolcke and Omohundro, 1994).

a S→ABABABAB (6) [0.86]
| ABACABAB (1) [0.14]

b S→ZZ (6) [0.86]
| XYZ (1) [0.14]
X→AB (27) [1.00]
Y→AC (1) [1.00]
Z→XX (13) [1.00]

c S→ZZ (7) [1.00]
Z→AB (27) [0.66]
| AC (1) [0.02]
| ZZ (13) [0.32]

d S→SS (20) [0.42]
| AB (27) [0.56]
| AC (1) [0.02]

Figure 4.2.: (a) Initial naive grammar. (b) After *Substituting* AB with X, AC with Y, and XX with Z. (c) After *Merging* (X, Y) to X. (d) After *Merging* (X, Z) to Z. (e) After *Merging* (S, Z) to S. Please note that uncertainties of symbols are not considered in this example.

4.2.1. Active Substring Discovery

In our framework, each terminal symbol represents a primitive action unit which contains a probability value, i.e. the symbol detector confidence. Each non-terminal symbol represents an abstraction of terminal symbols.

To generate a grammar that focuses on patterns with strong regularity, we build an n-gram-like frequency table which keeps the number of occurrences of substrings that are subset of input sequences. The score of a rule $X \rightarrow \lambda$ is the occurrence value of λ in the frequency table multiplied by the expected probability value of λ. Its calculation will be discussed in Section 4.2.2. This is different the method used in (Kitani, Yoichi, and Sugimoto, 2008) where they use a similar table to choose the best candidate symbols which has the maximum compression rate for *Substitute* operation.

For simplicity, we first consider the case without uncertainty values. In this case, as defined in (Stolcke and Omohundro, 1994) and (Kitani, Yoichi, and Sugimoto, 2008), the rule probability is calculated by normalizing rule

scores, i.e.:

$$P(X \to \lambda_i) = \frac{f(X \to \lambda_i)}{\sum_k f(X \to \lambda_k)} \tag{4.1}$$

where λ_i is the i-th rule production of non-terminal X and $f(\cdot)$ denotes the frequency of the string. $P(X \to \lambda_i)$ satisfies the following property:

$$\sum_i P(X \to \lambda_i) = 1 \tag{4.2}$$

In our method, as we keep counts for all possible sub-patterns from input samples, the probability of each rule is always larger than zero even if there was no input sequence that exactly matches the discovered sub-pattern. This has an effect of stronger "inductive leap", i.e. higher tendency to generalize from a relatively small number of input samples.

To illustrate, suppose that we want to learn a task with repetitions $(ab)^n$ from the 6 correct samples of "*abababab*" and 1 erroneous sample of "*abacabab*". The initial naive grammar (Figure 4.2(a)) simply contains all input sequences. We use parentheses $(\cdot)$ and brackets $[\cdot]$ to represent counts and probability values, respectively, e.g. $S \to ABC$ (20) [0.90] represents the rule score of 20 and rule probability 0.90. We now apply a *Substitute* (Figure 4.2(b)) and *Merge* operators (Figure 4.2(c)-(e)) introduced in (Stolcke and Omohundro, 1994) with rule scores obtained from our frequency table. Figure 4.2(a) shows an initial naive grammar. After *Substituting* AB with X, AC with Y, and XX with Z, we obtain the grammar in Figure 4.2(b). After $Merging$ (X, Y) to X, $Merging$ (X, Z) to Z, and finally $Merging$ (S, Z) to S, we obtain the grammar in Figure 4.2(e).

We have now obtained a more generalized grammar that favors (yielding higher probability when parsed) input sequences mostly containing AB's. It is worth noting that the rule probability of erroneous symbol AC is still

in the grammar but with very low probability. As a result, this grammar "allows" occasional errors as it still accepts noise cases with low probability instead of simply rejecting. This "soft" classification is one of the advantages of SCFGs, when compared to non-stochastic CFGs which do not have rule probability values.

In practice, it is often useful to limit the maximum length of symbols to be considered in the frequency table to avoid generating an exhaustive list of symbols to increase the speed. This is a reasonable assumption as human activities often involve repetitive action components(Zhou, Torre, and Hodgins, 2008). Also, considering only the n-most frequent substring patterns is an effective alternative. Since the search space of the possible grammars is not small, a beam search strategy is applied as in (Stolcke and Omohundro, 1994) which considers a number of relatively good grammars in parallel and stops if a certain neighborhood of alternative grammar models has been searched without producing further improvements.

4.2.2. Considering Input Samples with Uncertainty

So far, we have only considered a case where input symbols are non-probabilistic, i.e. terminals $(a, b, c...)$ are not assigned with probability values. However, since we assume that primitive action detectors could also provide uncertainty (confidence) values as output, it is beneficial to exploit this information. If there is a higher rate of noise, it is more likely that the certainty of a symbol is lower. Based on this assumption, we first compute the probability of a sub-pattern $\lambda = s_1 s_2 s_3 ... s_l$ of length l from input, as

$$P(\lambda) = (\prod_{i=1}^{l} P(s_i))^{\frac{1}{l}} \tag{4.3}$$

The term $\frac{1}{l}$ is used to scale the likelihood, since the probability is in overall multiplied by the number of symbols at the end of the parsing. The expected value of λ, $\mu(\lambda)$, is obtained by averaging all occurrences of $\lambda = \{\lambda_1, \lambda_2...\}$ in the input, i.e.:

$$\mu(\lambda) = \frac{1}{n}\sum_{k=1}^{n} P(\lambda_k) \tag{4.4}$$

Here, $\{\lambda_1, \lambda_2...\}$ are the same strings with possibly different probabilities, since each λ_i was sampled at different times by the detector. In prior works (Stolcke and Omohundro, 1994; Kitani, Yoichi, and Sugimoto, 2008), $\mu(\lambda)$ is computed directly by the occurrences of λ, whereas in our case, we take into account the confidence values of detectors. Thus, we modify the equation (4.1) as

$$P(X \to \lambda_i) = \frac{f(X \to \lambda_i)\mu(\lambda_i)}{\sum_k f(X \to \lambda_k)\mu(\lambda_k)} \tag{4.5}$$

where the subscript i denotes the i-th rule of X. We use this equation throughout our experiments. The above rule probability is used to compute the maximum likelihood in our Bayesian framework to update the MDL score (Equation 4.9).

In our method, we define the model prior probability

$$P(M) = P(M_S, M_\theta) = P(M_S)P(M_\theta|M_S) \tag{4.6}$$

where $P(M_S)$ and $P(M_\theta|M_S)$ respectively denote the structure prior and the parameter prior of a grammar. $P(M_S)$ is defined as Poisson distribution with mean (average production length) 3.0, as in (Stolcke and Omohundro, 1994; Kitani, Yoichi, and Sugimoto, 2008). The higher mean value means that the expected length of a production rule is larger.

In the literature, it is a common choice to model the parameter prior

$P(M_\theta|M_S)$ as a symmetric Dirichlet distribution for grammar induction, e.g. (Sakakibara, Brown, Hughey, Mian, Sjölander, Underwood, and Haussler, 1994; Shan, McKay, Baxter, Abbass, Essam, and Nguyen, 2004; Stolcke and Omohundro, 1994; Kitani, Yoichi, and Sugimoto, 2008). This is because Dirichlet prior is a conjugate prior, which allows the posterior distribution to be a simple product of the prior and the likelihood. The parameter prior is the product of Dirichlet distributions, each of which corresponding to the prior distribution over possible n expansions of a single non-terminal X:

$$P_X(M_\theta|M_S) = \frac{1}{\beta(\alpha_1, ..., \alpha_n)} \prod_{i=1}^{n} {\theta_i}^{\alpha_i - 1}, \tag{4.7}$$

where β, the normalizing factor, is a multinomial Beta function with parameters α_i, and θ_i is a rule prior which is uniformly distributed. Since we have no prior knowledge about the distribution of the parameters, $\alpha_i = \alpha_j$ $\forall i, j$ and $\sum_i^n \alpha_i = 1$.

Here, we briefly discuss about the effect of the values of α in a symmetric Dirichlet distribution, where $\alpha_i = \alpha_j$ $\forall i, j$, while computing the MAP estimates. If $\alpha_i > 1$, the resulting grammar tends to have rule probabilities that are more equally likely as α_i gets larger, even if the rule probabilities computed in Equation 4.5 are biased. If $\alpha_i < 1$, the rule probabilities tend to spread out towards extremes (0 or 1), where this tendency becomes stronger as α_i reaches towards zero ($\alpha_i > 0$). Lastly, if $\alpha_i = 1$, there is no prior on the distribution of rule probabilities, thus depending only on the rule probabilities computed by Equation 4.5.

We apply a pruning process as in (Stolcke and Omohundro, 1994) to speed up the induction and filter out non-critical production rules having probabilities lower than a certain threshold τ, as they are often accidentally

created due to noise. If the removal of a rule decreases the description length of model prior but increases that of data likelihood in relatively small amount, it will lead to a better (lower) MDL score. We set $\tau = 0.01$ in all of our experiments. However, we later experimentally show the pruning effect in Section 4.6, by varying the threshold value.

4.2.3. Measuring the Quality of a Grammar

Our goal is to find a grammar that is sufficiently simple yet expressive as pointed out by Langley et al.(Langley and Stromsten, 2000). In his work, a minimum-description length (MDL) principle is used to decide whether or not to merge states.

We denote $P(M)$ as *a priori* model probability, where M is a grammar model that includes structure priors $P(M_S)$ and parameter priors $P(M_\theta)$ that do not consider the input data D, where $P(D|M)$ denotes a data likelihood:

$$P(M) = P(M_S, M_\theta) = P(M_S)P(M_\theta|M_S) \tag{4.8}$$

where $P(M_S)$ specifies the structure prior, i.e. the length of a grammar, and $P(M_\theta)$ specifies the parameter prior, i.e. rule probabilities. Maximizing the joint probability $P(M, D)$

$$P(M, D) = P(M)P(D|M) \tag{4.9}$$

is equivalent to minimizing $-logP(M, D)$

$$-logP(M, D) = -logP(M) - logP(D|M) \tag{4.10}$$

where $-logP(M)$ represents the description length of the model under the

given prior distribution and $-logP(D|M)$ represents the description of the data D given a model M. The sum of two negative log values naturally corresponds to the total description length of the model and data. Thus, the goal can be rephrased as minimizing $-logP(M, D)$.

Although one can define the prior distribution of $P(M_S)$ in a simple form such as e^{-l}, where l = number of bits required to encode the grammar, it is far from being a natural distribution for grammars. Thus, a Poisson distribution is commonly used with a mean of 3.0 (average production length) as in (Stolcke and Omohundro, 1994) and (Kitani, Yoichi, and Sugimoto, 2008).

The data likelihood $P(D|M)$ is computed using Viterbi parsing, which is commonly used in HMMs. However, unlike (Stolcke and Omohundro, 1994) and (Kitani, Yoichi, and Sugimoto, 2008), to handle the uncertainty values of the input symbols, the method of computing the likelihood needs to be modified. To cope with this situation, we use the SCFG parsing algorithm with uncertainty input introduced in (Ivanov and Bobick, 2000) to compute data likelihood.

The following figure (Figure 4.3) summarizes the whole process.

4.3. Bag-of-Balls Experiment

4.3.1. Experiment Design

In this experiment, we assume a scenario where an arbitrary number of balls is put into a bag (denoted as a), moved to another place (denoted as b), and the same number of balls is taken out later (denoted as c), which can be represented in the form $a^n bc^n$. The samples are randomly generated from this model grammar up to the length of 9 (n=4).

1. Run symbol detectors on the input data to obtain action symbols, $\boldsymbol{\lambda} = \lambda_1, \lambda_2, ..., \lambda_n$.
2. Compute the substring frequency table (n-gram table) of the input symbols.
3. Initialize an empty search tree.
4. Construct a naive grammar and set it as the root node, $S \rightarrow \boldsymbol{\lambda}$ [1.0]
5. Apply operators defined in Section 4.2.1 and recompute rule probabilities.
6. Adjust rule probabilities according to symbol likelihoods as described in Section 4.2.2.
7. Add the newly acquired grammar as a child node and compute its MDL score.
8. Repeat steps 5-7 until no further improvement is found in terms of MDL score.

Figure 4.3.: Learning summary

To test over noise sensitiveness, we add *Insertion* and *Substitution* errors. An *Insertion* error inserts a random symbol into the input and a *Substitution* error randomly replaces a symbol with any incorrect one. We test with the noise probability in the range of [0%, 20%] with 1% step, totaling in 21 noise conditions. A noise probability of 10% means that either a *Substitution* or *Insertion* error has occurred in approximately 10% of the input symbols. Each noise condition is conducted 10 times with randomly generated dataset and its mean MDL score is computed, resulting in 210 experiments in total. We compare the results using our method and two previously reviewed methods proposed by Kitani (Kitani, Yoichi, and Sugimoto, 2008) and Stolcke (Stolcke and Omohundro, 1994). We also compute the MDL score ratio between the learned grammar and the hand-made model grammar.

The confidence values of terminal symbols are given such that the correct symbol is assigned with the probability computed from Gaussian distribu-

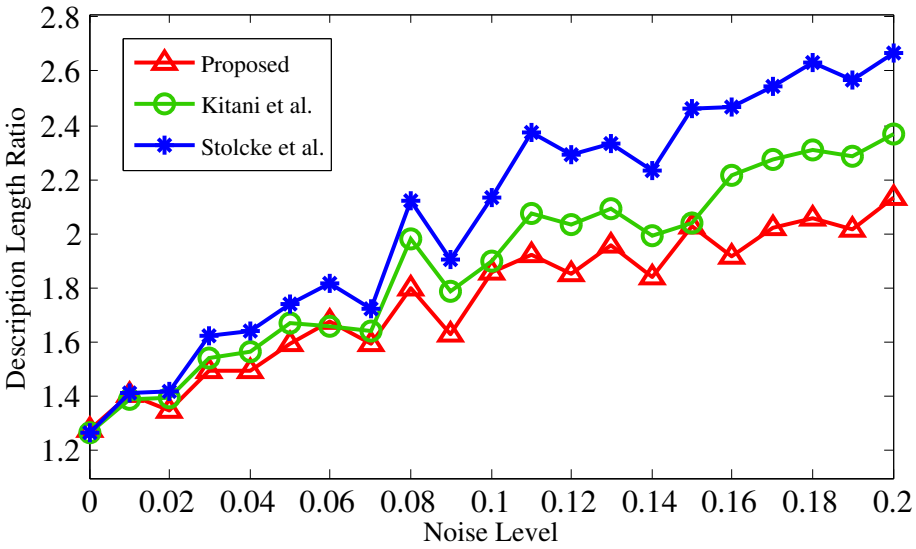


Figure 4.4.: Description length ratios of grammars generated by different methods. The lower score indicates that the grammar is more compact yet maintains sufficient expressive power.

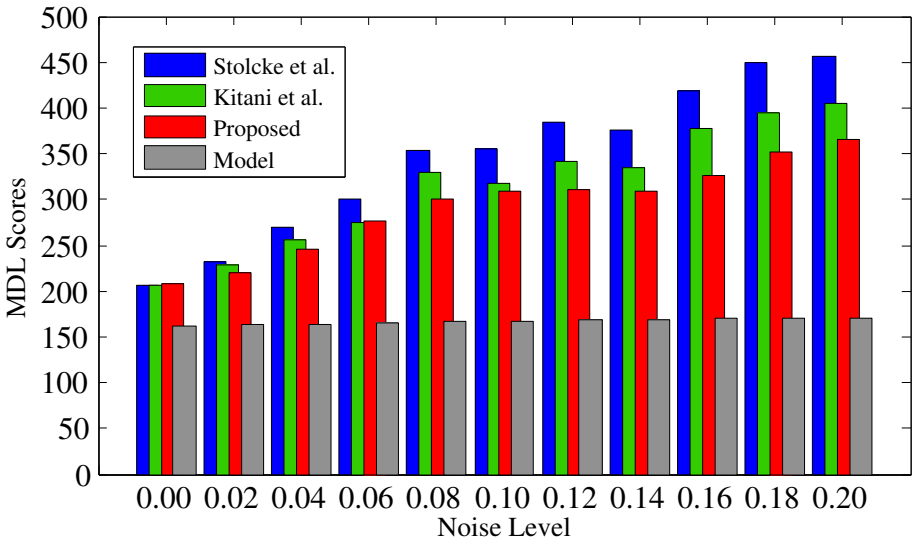


Figure 4.5.: Actual MDL scores for each method compared with the model grammar. MDL scores are averaged over 10 trials for each noise condition. The graph is shown with a 2% step for better view. A lower score indicates that the grammar is more compact yet reasonably expressive. How these scores affect the performance in the real world will be discussed in Sections 4.4 and 4.5.

(a)			(b)		
S→Y	(8.00)	[0.53]	S→AABCC	(6.99)	[0.60]
| AYC	(3.00)	[0.20]	| ASC	(2.66)	[0.23]
| AABAC	(1.00)	[0.07]	| AASCC	(0.93)	[0.08]
| AACACCCC	(1.00)	[0.07]	| CS	(0.64)	[0.05]
| AAYCC	(1.00)	[0.07]	| AABAC	(0.46)	[0.04]
| CY	(1.00)	[0.07]			
Y→AABCC	(8.00)	[1.00]			

Figure 4.6.: The obtained grammars using the method in (Kitani, Yoichi, and Sugimoto, 2008)(a) and the proposed method(b) from data with noise probability 0.08.

tion with $\mu = 0.85, \sigma = 0.1$ and wrong symbol with $\mu = 0.15$, $\sigma = 0.1$. We set unrelated symbol d to be included as noise, as in (Kitani, Yoichi, and Sugimoto, 2008).

The description length ratio of a grammar is the ratio of MDL scores between learned grammar and the model grammar, where the lower score indicates that the grammar is more compact yet maintains enough expressive power. Figure 4.4 shows description length ratios over various noise conditions, where in most cases the grammars generated by our proposed method have the lowest description length ratio implying that they are well-balanced between compactness and expressiveness. We prune production rules that are less than 1%, which are generally obtained due to noise.

4.3.2. Findings

As qualitative analysis, we now examine some of the obtained grammars. In the case with noise probability 0.08, a grammar obtained using the method proposed in (Kitani, Yoichi, and Sugimoto, 2008) is shown in Figure 4.6(a). Under this noise condition, the mean MDL score was 330.38 and the standard deviation was 39.72. A grammar obtained using our proposed method under the same noise condition with the same dataset is shown in Figure

4.6(b). The mean MDL score was 300.62 and the standard deviation was 48.27. The average MDL scores can be seen in Figure 4.5.

It is worth noting that the rule scores in the grammar generated using our method reflect the uncertainty values of input symbols. As a result, in Figure 4.6(b) the erroneous sequence $AABAC$ (the last rule) has a rule score of 0.46 in contrast to 1.00 in Figure 4.6(a), as the symbol C had lower probability (higher uncertainty) due to noise. In the second grammar, since rules containing noise quickly converged to very low probability (less than 0.01) and pruned, the rule probability for the correct cases, e.g. $S \rightarrow AABCC$ has a relatively higher probability value. This will result in higher likelihood when parsed on new samples within the same class.

In the following sections, we show how MDL scores actually reflect the performance in several real world robot scenarios.

4.4. The Towers of Hanoi Experiment

The aim of this experiment is to make a robot correctly recognize and imitate the human task sequences for successfully executing a task. However, instead of simply imitating, we require that the robot should deal with noise using the previously obtained knowledge so that it can perform the intended task sequence correctly even when the perceived actions are partially incorrect. Furthermore, we are interested in challenging tasks that include recursion which can be demonstrated with various lengths of task sequences. We choose the Towers of Hanoi problem as it satisfies the above requirements. As discussed in Section 4.1, we tackle the problem from the "*what* to imitate" perspective, i.e. at the symbolic level rather than trajectory level. Thus, it is worth mentioning that in this experiment we represent

each action symbol as an action goal rather than its trajectory.

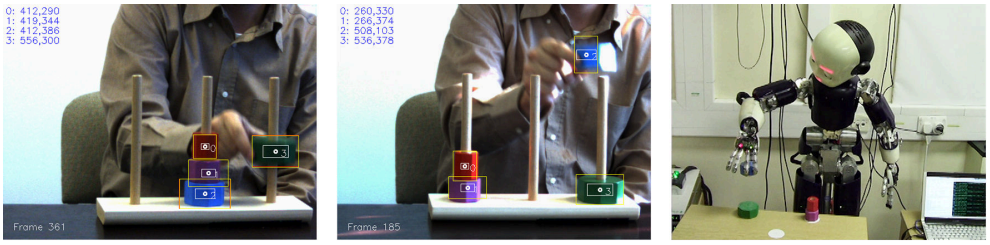
4.4.1. Experiment Design

We evaluate our method on real-world data obtained from the demonstrations of 5 human participants. In the training phase, a human demonstrator shows solving the puzzle using 2 and 3 disks, respectively, repeating each task 3 times. The robot then learns a task grammar from each demonstrator using techniques explained in Section 4.2. Thus, 5 task grammars are learned in total.

In testing phase, a human demonstrator solves the puzzle using 4 disks, repeating 3 times. The trained task grammar is used to parse the observation, which generates a sequence of primitive actions to execute. A reproduction is considered a success only if the robot solves the puzzle by correctly executing the complete sequence of primitive actions. Each task grammar is used to parse each demonstration, which results in 15 tests for each of our 5 participants, or 75 in total. We use iCub (Metta, Sandini, Vernon, Natale, and Nori, 2008), a humanoid robot with 53 degree of freedom, as our testing platform. Figure 4.7(c) shows a sample image of iCub executing the parsed primitive actions.

We experiment under two types of noise conditions: the low-noise (indoor lighting) and high-noise (direct sunlight) conditions. That is, a) train on the low-noise condition and test on both low- and high-noise conditions, respectively, and b) train on the high-noise condition and test on both conditions. All samples of the high-noise data set were captured in the same day for consistency. Example samples can be seen in Figure 4.7.

Since we are interested in high-level task representations, we assume that the system can detect minimal level of meaningful primitive actions and



(a) (b) (c)

Figure 4.7.: (a-b) A sample tracking screen while a human participant is solving the puzzle with 4 disks. Compared to the low-noise condition (a), the high-noise condition (b) shows overexposed spots which often makes the tracker unstable. The tracker immediately resets the position if lost by searching the desired blob from the entire region of the image. (c) shows iCub performing parsed primitive actions. A demo video is available at: http://www.youtube.com/watch?v=S99ViThK050

generate symbols. Similar to (Ivanov and Bobick, 2000), we define these primitive action detectors using HMMs where each model corresponds to an action symbol with its output value representing the symbol's certainty, or probability value. The input to these detectors are the currently moving object's quantized direction, and distances between the object and towers.

In this experiment, our system generates 5 types of primitive action symbols during an observation as detailed in Figure 4.8. The reason we define symbols like *Disk moved "between" A and B* instead of *Disk moved "from" A to B* is because they are sufficient to represent the task structure without generating an excessive number of symbols. As the rule of the puzzle enforces that only a smaller disk shall be placed on top of the bigger disk, there is always only a single possibility of moving a disk between two towers. This is a fair assumption as this rule is always given in prior, not learned. Thus, in terms of executing symbols A, B, and C, we can expect that the robot will make the correct move. During the training phase, the symbol with the highest certainty is fed into the input of the grammar building

Symbol	Actions
L	Lift a disk from the tower
D	Drop a disk into the tower
A	Move between tower 1 and tower 2
B	Move between tower 1 and tower 3
C	Move between tower 2 and tower 3

Figure 4.8.: Primitive actions defined in *Towers of Hanoi* experiment. The system is equipped with these 5 primitive action detectors which generates symbol probability during observation.

algorithm.

If we denote action sequences LAD as X, LBD as Y, and LCD as Z, then symbols X, Y, and Z represent `pick-and-place` action sequences. The optimal solution of the puzzle can be represented as $((LAD)(LBD)(LCD))^n$, or $(XYZ)^n$, meaning "Perform (XYZ) recursively until the problem is solved."

Object trackers are implemented using standard CamShift algorithm provided in (Bradski, 2000), with additional Kalman filtering to improve stability. A sample tracking screen is shown in Figure 4.7; as it depends on the color information of blobs, it often produces errors due to lighting conditions.

We use the standard Cartesian control library developed by Pattacini et al. (Pattacini, Nori, Natale, Metta, and Sandini, 2010) and a grasp trajectory planning method reported in (Su, Wu, Lee, Du, and Demiris, 2012) to execute the Tower of Hanoi task on iCub. We use this method to effectively deal with position errors of disks, which internally uses a grasp simulator to plan the optimal trajectory of hand joints for every disk. The advantage of this method is that it can cope with arbitrary shapes and sizes of disks.

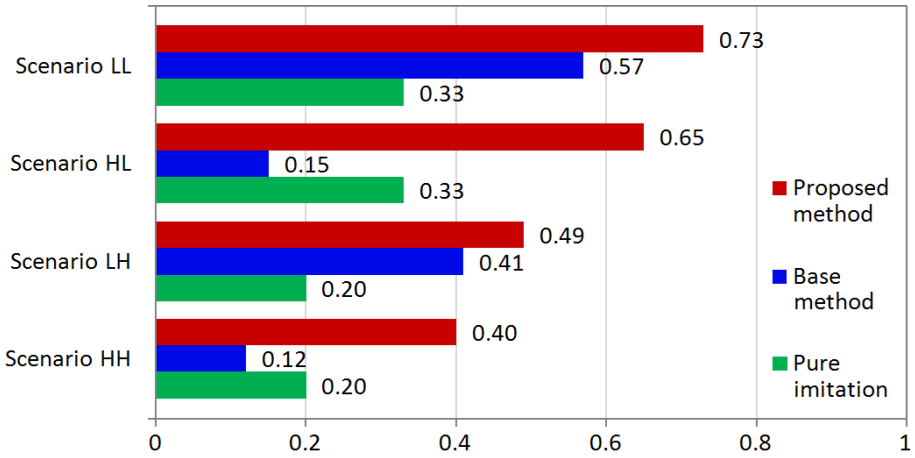


Figure 4.9.: Success rates using our method, base method and the pure imitation. Scenarios LL and LH: Train on the low-noise condition and test on low- and high-noise conditions, respectively. Scenarios HL and HH: Train on high-noise condition and test on low- and high-noise conditions, respectively. The fact that a single mistake while parsing a long test sequence causes a failure makes this problem non-trivial.

4.4.2. Findings

As explained in the last section, the objective here is to learn a high-level task representation from a few short sequences of demonstrations that can be used to better parse unforeseen, possibly more complicated tasks that share of same task components. We report the performances in 4 scenarios (LL, LH, HL, HH) in Figure 4.9.

In scenarios LL and LH, models are both trained from demonstrations of 2 and 3 disks under the low-noise condition, then they are tested on demonstrations of 4 disks on the low-noise and high-noise conditions, respectively. Similarly, scenarios HL and HH are trained from the high-noise condition and tested on both noise conditions.

We compare with the base method (Stolcke and Omohundro, 1994) and the pure imitation method which simply follows what has been observed

Scenario	Method	Success	Avg.MDL	Scenario	Method	Success	Avg.MDL
LL	Proposed	55	284.63	LH	Proposed	37	286.92
	Base	43	390.28		Base	31	393.26
	Pure Imi.	25	N/A		Pure Imi.	15	N/A
HL	Proposed	49	306.25	HH	Proposed	30	306.66
	Base	11	469.32		Base	9	469.46
	Pure Imi.	25	N/A		Pure Imi.	15	N/A

Figure 4.10.: Detailed results with average MDL scores for comparison. Each case is tested on 75 sequences. MDL score is not available for the pure imitation as it does not rely on any learned model. It is worth noting that lower MDL scores generally lead to higher success rates.

Demonstrations using 4 disks	Low-noise	High-noise	Total
Total number of sequences	15	15	30
Sequences containing wrong symbol	10	12	22
Average number of error symbols per trial	1.13	2.20	1.67

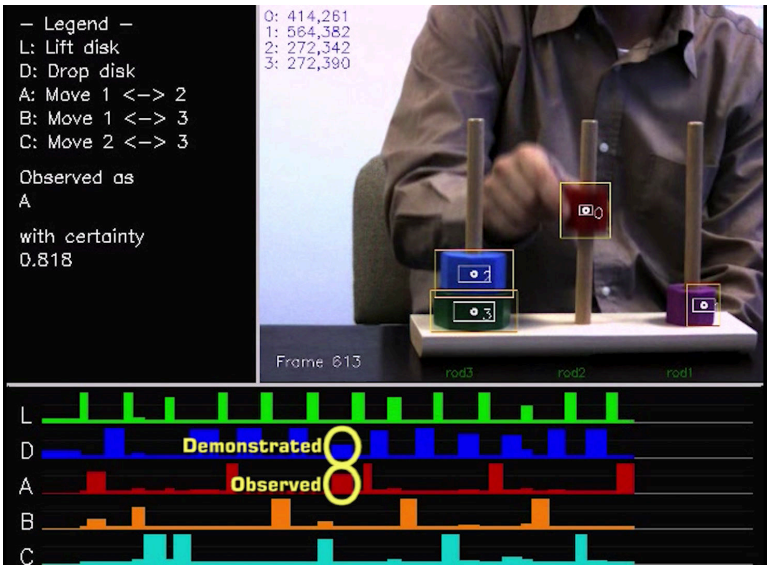
Figure 4.11.: Error statistics of demonstrations using 4 disks on each noise condition. Note that even in the low-noise condition, there are only 5 trials observed with all correct symbols, which means that in most cases the pure imitation will not lead to the desired goal state. Each testing sequence is composed of 45 primitive action symbols, which makes this problem non-trivial as only a single mistake will make it fail to achieve the goal.

from demonstrations. In any case, if the system makes any single mistake while recognizing human demonstration due to either wrong tracking or wrong symbol interpretation, it is marked as failed. This makes our scenarios non-trivial as each testing sequence is composed of 45 symbols. Please refer to Figure 4.11 to see error statistics. We do not use the method proposed by Kitani et al(Kitani, Yoichi, and Sugimoto, 2008) in this experiment as all generated symbols are always related to the task.

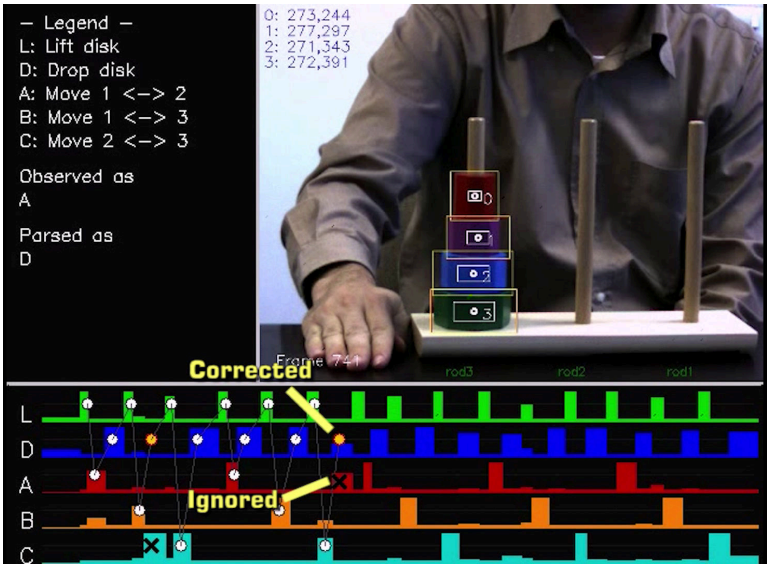
As can be seen in Figure 4.9, it is important to note that there is a noticeable difference on the base method between scenarios LL and HL, and between LH and HH. As scenarios HL and HH are trained from noisy training data, the task representations could be easily corrupted. This could even lead to parse the correct symbol into wrong symbol which results in worse performance than purely imitating observed actions, whereas our method at least performs better than the pure imitation.

Figure 4.12 shows a test example with 4 disks, where some of the ambiguous observations are clarified using the learned grammars at the parsing time. Figure 4.12a shows where the block is being dropped (symbol D). Due to tracker error, the certainty of symbol A was higher than symbol D. It was disambiguated and corrected at the parsing time, as shown in Figure 4.12b.

It is also worth noting that from Figure 4.10, we can confirm that lower MDL score leads to generally better representations. A model with the highest MDL score 469.46 (scenario HH, Base method) had the poorest performance, where a model with the lowest MDL score 284.63 (scenario LL, Proposed method) exhibited the best performance. As expected, models learned in the high-noise condition tend to have lengthier descriptions, which increases prior score. Relatively high MDL scores generally mean that they



(a) A participant demonstrates solving the puzzle using 4 disks, where the block is being dropped (symbol D). Due to tracker error, the certainty of symbol A was higher than D.



(b) Symbols that are inconsistent with the learned grammar are corrected at the parsing time.

Figure 4.12.: A test experiment scenario where 4 disks are used, requiring 45 actions to be correctly executed to reach the goal.

a
S→LAD [0.205669]
| LBD [0.204606]
| LCD [0.163233]
| CADSS [0.020551]
| SLBAS [0.017184]
| SSS [0.388758]

b
S→LADLBDLCD [0.666667]
| SSLAD [0.285714]
| SSSSS [0.047619]

c
S →LADLBDX [0.601507]
| LXCX [0.248180]
| SSLAD [0.150313]
X→ADLL [0.200956]
| LCD [0.799044]

Figure 4.13.: (a) A sample grammar that captured the meaningful task components such as LAD, LBD, and LCD, which can be used to enforce the observation to be consistent with the demonstrator's intended actions. $CADSS$ and $SLBAS$ come from occasional noisy examples and hence they are assigned very probabilities. (b) A grammar learned from an ideal (noise-free) dataset. (c) A grammar learned from the same dataset of (a), but with a pruning threshold of 0.15. Please see Section 4.6 for more detailed analysis on pruning effects.

are too specific, failing to capture the recursiveness nature of the task.

The example grammar constructed using the proposed method (Figure 4.13(a)) shows that it captured meaningful task components: LAD, LBD, and LCD. (lines 1-3) Task components $CADSS$ and $SLBAS$ come from occasional noisy examples and hence they are assigned very probabilities. Although there are intermittent error symbols in input sequences, the underlying structures of task components are captured effectively. The knowledge of these underlying structures allow to filter out contextually inconsistent observations. For example, the learned task component LAD allows the action DROP (D) to be expected when MOVE BETWEEN (A) action is observed, even if DROP action was missed or misinterpreted. The last line of the grammar rules shows that it also captured the recursiveness nature of the task.

Although each model is constructed only from 6 sample sequences, it

successfully captured these core components due to active substring searching explained in Section 4.2.1. Figure 4.13(b) shows an example grammar constructed from data that contains no noise at all. Most of the experiments, however, include noise symbols in the middle of an input sequence which hinders the discovery of the full meaningful task component such as $LADLBDLCD$ in Figure 4.13(b), line 1. Nevertheless, grammars discovered like the one in Figure 4.13(a) worked reasonably well to support parsing the same task with more complicated sequences.

It is worth mentioning that although human participants were given the optimal solution in prior, 75% of the participants made one or more mistakes due to confusion while solving when the number of disks was 4.

4.5. The Dance Imitation Experiment

4.5.1. Experiment Design

In this experiment, we define 3 types of dance demonstrations. The goal of this experiment is to learn the generalized representation of human dance movements, which is utilized to recognize more complex movements. Each dance sequence is composed of a subset of predefined motion primitives, i.e. dance symbols.

The input to the system are time-series 54-dimensional angular values of 18 human joints, captured using an OptiTrack 8-camera motion capture system. Temporal segmentation is applied (Section 4.5.1), where each segment is mapped to one of 9 primitive dance symbols. To map segments to symbols, we need to train detectors (Section 4.5.1). After obtaining detectors, we can now convert a video stream into a sequence of symbols which is fed into our SCFG learning framework. Finally, the robot performs dance

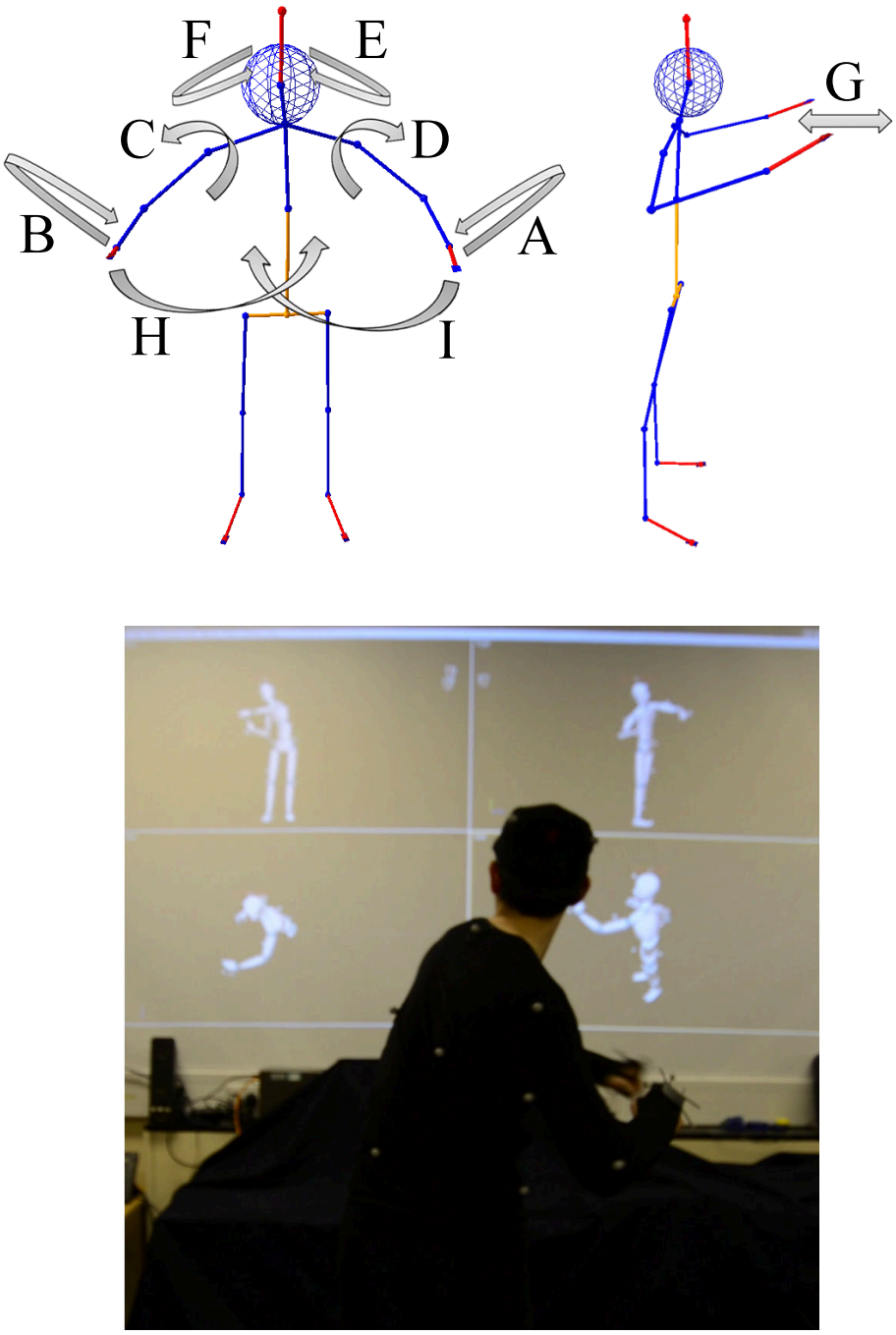


Figure 4.14.: 9 motion primitives used in this experiment and a demonstration example. Please see the following video for better visualization: http://www.youtube.com/watch?v=S99ViThK050.

Grammar	$(CD)^n(EF)^n$	$(ABE)^n$	H^nGI^n
Train set	n=1,2	n=1,2,3	n=1,2,3
Test set	n=3,4	n=4,5	n=4,5

Figure 4.15.: 3 types of dance representations used in the experiment. Please see Figure 4.14 for reference. In training set, there are 5 trials for each value of n (sequence length), which results in 40 dance demonstrations (225 input symbols). The testing set has 6 trials for each n, which results in 36 dance demonstrations (450 input symbols).

primitives by executing the parsed symbols. We map the human joints into iCub's joints and generalize the trajectories of 9 motion primitives from multiple demonstrations. (Section 4.5.1)

The dance grammars used to generate data samples in this experiment are: 1) $(CD)^n(EF)^n$, 2) $(ABE)^n$, and 3) (H^nGI^n). We describe the scenario settings in Figure 4.15.

Temporal Segmentation

We modify the temporal segmentation method proposed by Fod et al. (Fod, Mataric, and Jenkins, 2002) which segments human motions at zero-crossing points of the squared sum of joint velocities. Similar to (Fod, Mataric, and Jenkins, 2002), where they selected a subset of joints, we select four sets of human joints (usually between 3 and 5 out of 18) that move significantly in

Joints Set	Involved Human Joints	Motion Primitives
Left arm	Chest, Left shoulder & Elbow	A, D, I
Right arm	Chest, Right shoulder & Elbow	B, C, H
Two arms	Chest, Left and Right shoulder & Elbow	G
Head	Chest, Neck, Head	E, F

Figure 4.16.: The informative human joints chosen to be used for calculating the ASV and ASD values. As these joints are often overlapped across multiple motion primitives, the number of the joint sets are reduced to four.

every motion primitive, as shown in Figure 4.16. Furthermore, we compute two types of features for segmentation: the average of squares of joint velocity (ASV, Eq. 4.11) and the average of squares of joint distance to the initial posture of the dance sequence (ASD, Eq. 4.12).

$$ASV(\mathcal{S}, \omega) = \sum_{i \in S} \omega_i^2 / Card(S) \tag{4.11}$$

$$ASD(\mathcal{S}, \theta, \theta^r) = \sum_{i \in S} (\theta_i - \theta_i^r)^2 / Card(S) \tag{4.12}$$

where $\mathcal{S}$ is the set of joints as defined in Figure 4.16, ω_i is the velocity of joint i, $Card(\mathcal{S})$ is the cardinal number of $\mathcal{S}$, θ_i is the position of joint i, and θ^r is the vector of joints position of the reference posture.

For each time step, we choose $\mathcal{S}$ with the largest ASV value for segmentation. Then we find the zero crossings of the ASV where ASD value is lower than a threshold. In our case, the threshold is automatically computed from data by clustering ASD values into two groups and taking the mean of two cluster centers. We use K-means (K=2) for clustering. An example is shown in Figure 4.17.

Training of Symbol Detectors

After obtaining video segments, we first compute the angular velocity of joints by computing the frame differences of 54-dimensional joint data, followed by taking Bag-of-Words (BoW) approach combined with one-vs-all SVM. We cluster the joint velocity data into K clusters using K-means (K=50), and use them to compute the histogram of every segment. One-vs-all multiclass SVM classifiers are trained from these histograms for 9 different symbols using radial basis function (RBF) kernel, similar to (Barnachon,

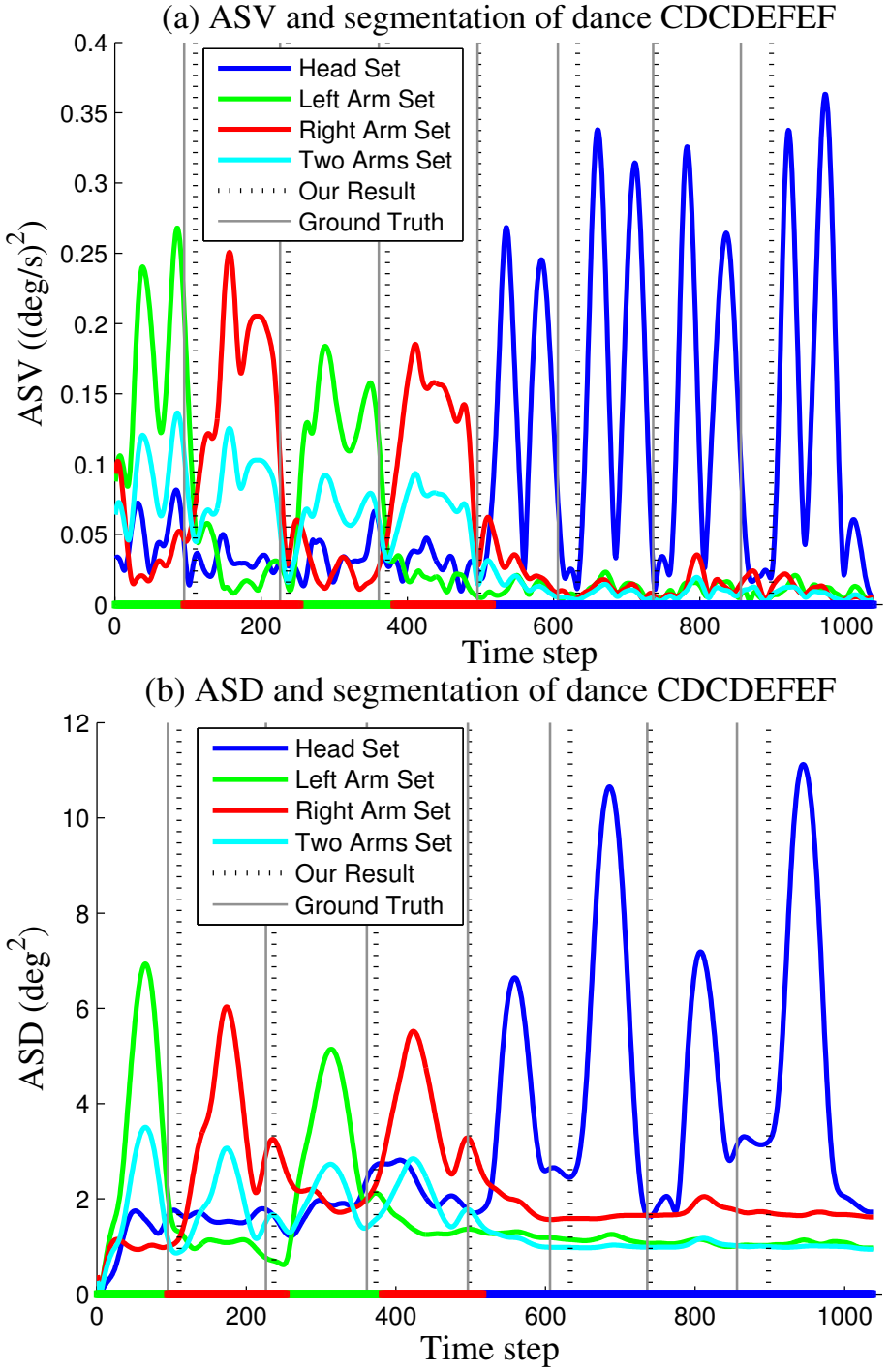


Figure 4.17.: The ASV(a) and ASD(b) of the movement sequence: the used joints set for each time step is marked on the bottom using corresponding color. The zero-crossings of ASV with sufficiently low ASD value are chosen as the segmentation points.

Bouakaz, Boufama, and Guillou, 2014). We use LibSVM library (Chang and Lin, 2011) to train and test SVMs. After running a grid search optimization, we obtained RBF kernel parameters of $C = 0.5$, $\gamma = 0.0078125$.

Trajectory Generalization

After classifying each segmentation, we use the segments that belong to the same class as the training set to generalize the trajectories for iCub. Dynamic Time Warping (Chiu, Chao, Wu, Yang, and Lin, 2004) is applied to demonstration sets to gain trajectories for each motion primitive, which are then mapped to the corresponding joints of iCub. The joint configurations of iCub's chest and head are the same as those of human, which makes it possible to directly assign the angles of these joints to iCub. But the configurations of iCub's arm and the human arm are different, so we map these joint angles to the iCub by minimizing the error of the directional vectors of the upper and the lower arm between the human and iCub under the constrains of the joint limits of iCub's arm. Now iCub is ready to execute the sequence of dance symbols. Figure 4.18 shows the representative frames of one of 3 dance sequences.

4.5.2. Findings

Figure 4.19 shows the performance in 4 scenarios, similar to the Towers of Hanoi experiment in Section 4.4. We define the low-noise condition (L) when the ground-truth segmentation is used, and the high-noise condition (H) when automatic segmentation described in Section 4.5.1 is used. The first letter corresponds to the training condition, whereas the second letter corresponds to the testing condition. For example, "LH" means the grammar was learned using manually segmented sequences from the training

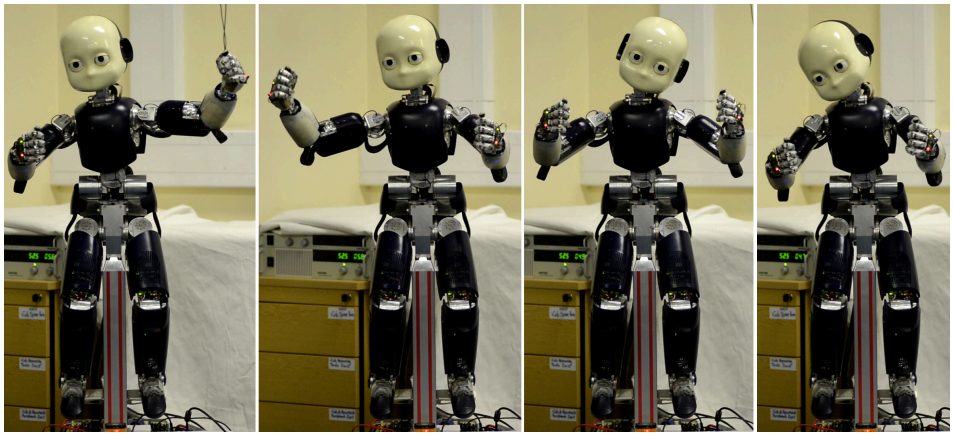


Figure 4.18.: iCub performing parsed primitive actions. Each figure from the left to right respectively represents primitive actions C, D, E, and F. The video containing full movements can be seen on: http://www.youtube.com/watch?v=S99ViThK050.

dataset, and parsed on automatically segmented sequences from the testing dataset. Since there are a significant number of input error symbols, we also denote the actual number of symbols that are recognized correctly. In the pure imitation (no grammar) case, the number of correct symbols are equivalent to the number of correctly recognized symbols by symbol detectors.

Figure 4.20 shows the learned grammars of 3 dance representations from the demonstrations using automatically segmented sequences as training dataset computed by the method described in Section 4.5.1. This training dataset is marked as the high-noise case (H) since the higher error in the segmentation leads to a higher error rate on the symbol detection, which affects on grammar learning. Thus, these grammars are used to test scenarios "HL" and "HH".

Figure 4.21 shows the learned grammars using manually segmented sequences as training dataset. It is notable that only the segmentation part was done manually. The training and testing of symbol detectors and gram-

Scenario	Method	Correct	Success	Avg.MDL
LL	Proposed	450	36	400.09
	Base	450	36	408.70
	Pure Imi.	437	30	N/A
LH	Proposed	450	35	413.63
	Base	450	35	422.26
	Pure Imi.	347	11	N/A
HL	Proposed	450	36	450.99
	Base	414	24	464.93
	Pure Imi.	437	30	N/A
HH	Proposed	424	30	464.27
	Base	414	24	476.12
	Pure Imi.	347	11	N/A

Figure 4.19.: Detailed results with average MDL scores for comparison. Each scenario has 36 sequences, and the total number of symbols per scenario is 450. "Correct" column shows the number of correctly recognized symbols after parsing, where in pure imitation case it is equivalent to the number of action detector errors. MDL score is not available for the pure imitation as it does not rely on any learned model. It can be seen that the lower MDL scores generally lead to higher success rates.

mar learning parts are all done in the same way as in the automatically segmented dataset. These grammars are used to test scenarios "LL" and "LH".

In Figure 4.19 (HL scenario), it can be seen that the pure imitation has a better performance than using grammars obtained using the baseline method. This is because of the high level of noise in the input hinders building a correct representation in the grammar. As a result, it sometimes leads to an adverse effect where the correct input symbols are identified as wrong. Our proposed method is less likely to suffer from this problem because the uncertainty values of input symbols and substring frequencies are considered.

The grammars shown in Figures 4.21(a) and (b), Figures 4.20(b) and (c)

(a)

S→EF [0.293200]
| SS [0.287079]
| CD [0.198005]
| SSSS [0.085637]
| CF [0.044922]
| CES [0.028048]
| CHS [0.024241]
| SFE [0.019778]
| SCIHFS [0.019089]

(b)

S→ABE [0.592059]
| SS [0.390003]
| SAAB [0.017939]

(c)

S→HGI [0.523234]
| HSI [0.415843]
| HESII [0.034387]
| HSG [0.026536]

Figure 4.20.: Acquired grammars from automatically segmented dataset using the method described in Section 4.5.1. The error in the segmentation leads to a higher error rate on detectors, which is regarded as the high-noise scenario.

(a)

S→CDEF [0.667192]
| CDSEF [0.332808]

(b)

S→ABE [0.598758]
| SS [0.401242]

(c)

S→HS [0.307153]
| SI [0.259863]
| HSI [0.257960]
| HG [0.144169]
| SG [0.020607]
| SF [0.010248]

Figure 4.21.: Learned grammars from manually segmented dataset, noted as the low-noise scenario. Note that only segmentation was done manually, where symbol detectors are still trained and tested in the same way as in automatically-segmented dataset.

actually captured the original grammar used to generate dance sequences, although the last one contains some unrelated symbols due to the higher level of symbol detector errors. They can effectively correct the wrong symbol patterns that largely differ from the symbol patterns in training sequences. Still, it is interesting to see that other two grammars partially capture the important constraints such as "HSI" and "HG" in Figure 4.20(c) and "EF" and "CD" in Figure 4.21(a).

For the execution of motion primitives, we concatenate learned trajectories of joints based on parsed symbol sequence and apply a low-pass filter to avoid discontinuity between symbols. Since all trajectories are learned from multiple human demonstrations, iCub can show natural human-like movements, such as subtle movements of torso and head while reaching an arm forward. A video of a demonstrator example can be found at: http://www.youtube.com/watch?v=S99ViThK050

4.6. The Effect of Pruning Factors

In this section, we show how the change of pruning thresholds affect the result. The range of thresholds are from 0.00 to 0.20, with intervals of 0.01. Figures 4.22 and 4.23 show how the pruning threshold affects the testing accuracy and grammar induction time. In this experiment, we train from all training data, i.e. all samples from both low-noise and high-noise conditions, and test on all testing samples as well. As in the previous experiments, a trial is regarded as fail even if there was a single error in the parsed symbols.

The result in Figure 4.22 confirms that although overall MDL score decreases as the threshold increases, the resulting grammar loses generality and shows poor performance on testing data. In Figure 4.23, training time

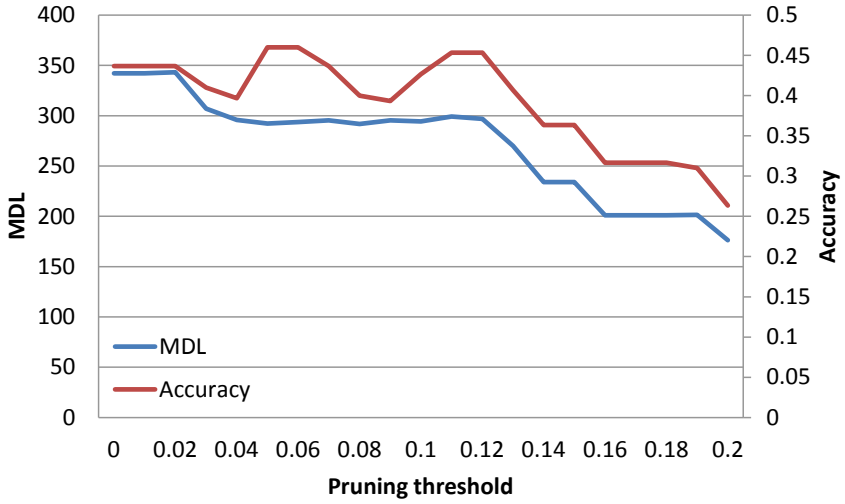


Figure 4.22.: The effect of different pruning parameters. In this experiment, we trained from all training data, i.e. all samples from both low-noise and high-noise conditions, and similarly tested on all testing samples. It can be seen that although overall MDL score decreases as threshold increases, the resulting grammar loses generality and shows poor performance on testing data. As in the previous experiments, a trial is regarded as fail even if there was a single error in parsed symbols.

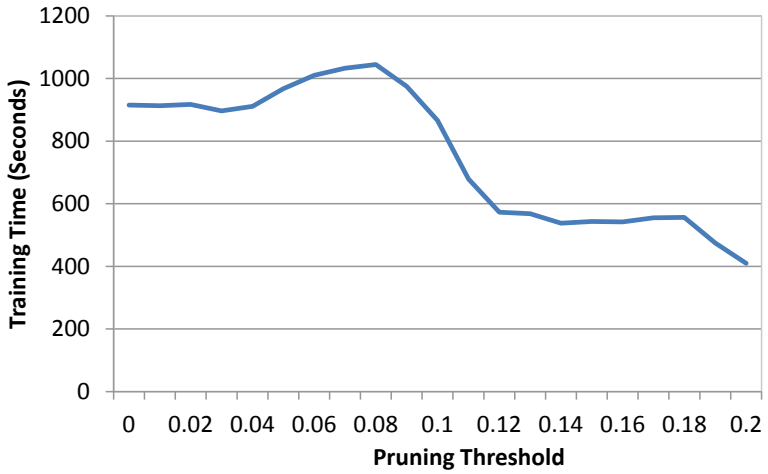


Figure 4.23.: The comparison of training times over different prune parameters. Since rules are more likely to be pruned as the threshold increases, the overall learning time tends to decrease. It was tested on a Linux desktop with i7 3.2GHz CPU, 16GB RAM, Python 3.2.

generally decreases as the pruning threshold increases, as more rules are more likely to be discarded during the induction process.

4.7. Summary

This chapter presents a robot imitation learning framework using probabilistic task grammars. Our method aims to discover reusable common task components across multiple tasks from input stream. The results in the two non-trivial real-world experiments (Sections 4.4 and 4.5) show that our method is capable to learn reusable structures under reasonable amount of noise, as well as in the synthetic dataset experiment (Section 4.3). [1]

In the Dance Imitation experiment (Section 4.5), the robot not only generalized the task from multiple demonstrations at the symbolic level, but also at the trajectory level, which makes our framework more complete. We have also experimentally shown that a lower MDL score generally leads to higher performance on parsing unforeseen action sequences.

The discovery of important task actions and recursions are critical to the performance, which is supported by the results reported in Sections 4.4.2 and 4.5.2. For example, the task component LAD in Figure 4.13(a), line 1 (Lift a disk, Move between towers 1 and 2, Drop) provides local constraints that enforce contextually consistent interpretation by biasing the parser to parse in the order of $L - A - D$ even when the observed symbols are partially wrong. This biasing effect can be also interpreted as an affordance learning, similar to (Lopes and Santos-Victor, 2005), where the recognition of an observed gesture depends on a context variable. Using the learned grammar in Figure 4.21(a), when the robot observes CD actions several

[1] The experiments in this chapter appear in (Lee, Su, Kim, and Demiris, 2013; Lee, Kim, and Demiris, 2012b).

times, it can "expect" to observe the same number of EF actions, which act as a belief system. Due to this advantage, wrong or uncertain symbols were often corrected or clarified by the learned grammar, e.g. Figures 4.12a and 4.12b.

The results reported in Section 4.3 support our idea that handling uncertainty values of input symbols improves the performance. Also, the human-readable results, e.g. Figures 4.13, 4.20, 4.21, is another benefit point in human-robot interaction domain.

We have shown in Section 4.6 how the pruning threshold affects the overall performance. Although we have used a fixed pruning factor for all experiments, it would be an interesting work to find an optimal parameter in a more systematic way, e.g. cross-validation within training samples. This will lead to a more compact representation of tasks while keeping the training time to minimum.

In the *Bag of Balls* (Section 4.3) and *The Towers of Hanoi* (Section 4.4) experiments, we have used 3 and 5 primitive symbols, respectively. While these were simple enough to show how our method works, *Dance* experiment scales up to 9 primitive symbols, which are similar to other real-world settings, e.g. 10 primitive symbols used to model complex employee-customer transaction activities in a convenience store (Kitani, Yoichi, and Sugimoto, 2008), 10 primitive symbols used to model car-human interaction scenarios in surveillance videos (Ivanov and Bobick, 2000), and 12 primitive symbols to model the Black Jack card game (Moore and Essa, 2002). The scalability of these methods to even more complex datasets that will necessitate even higher number of symbols remains an open challenge.

In *the Towers of Hanoi* experiment, we had an assumption where the robot knows that only a smaller object should be placed over larger object,

similar to how humans tell others when they give instructions on solving this specific puzzle. However, it would be an interesting work to make this more general, by making a robot to learn this rule well by using symbolic-level planners, such as STRIPS-like symbolic planners (Fikes and Nilsson, 1972).

The inclusion of domain-dependent, biased structural priors could be also beneficial in terms of both searching speed and grammar accuracy as certain models will be effectively rejected even if they retain good MDL scores. This will be especially useful in the domain of imitation learning which often shares many reusable components across different tasks.

In the following chapter, the study of automatically learning primitive action detectors in unsupervised manner will be presented.

5. Learning Action Components from Demonstrations

5.1. Introduction

The design of primitive action detectors are critical to task-level LfD approaches. So far, we have assumed that these primitive action detectors are given in prior to the experiment. But what if we have no prior knowledge about primitive actions that constitute a task? This chapter presents an unsupervised learning approach of primitive action symbols from human demonstrations, which self-tunes the number of action detectors required to represent the given hierarchical task effectively.

5.2. Automatic Discovery of Primitive Action Detectors

Given an input the unlabeled time-series signal segments, our goal is to discover a meaningful set of action symbols that can effectively represent a task. If we define too large number of symbols, the description complexity will increase, possibly lacking the generality of the task representation as well as it will capture all the subtle differences of human movements. On the other hand, if we define too small number of symbols, it will over-generalize

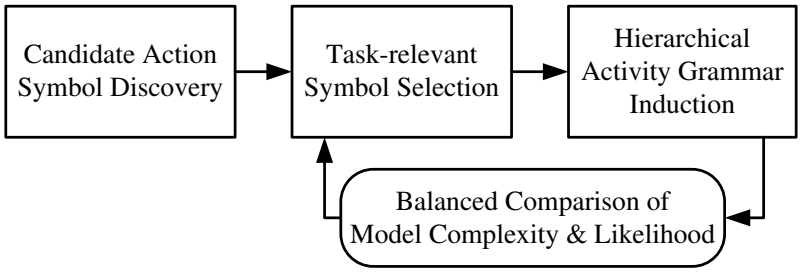


Figure 5.1.: Overview: Candidate symbols are generated using agglomerative hierarchical clustering approach, where too general or specific symbols are subsequently filtered out by measuring the model complexity and likelihood.

the task representation and result in the failure of capturing the meaningful differences of task components. Hence, it is useful to find a balancing point using the minimum description length of the induced grammar.

In our approach, we first discover a number of candidate "systems" where each system has a different set of action symbols. Next, we learn the SCFG representation using the method described in Chapter 4 for every candidate model, which feedbacks a model description length and likelihood value that are used to select models. Since the value ranges of prior probability and likelihood differ in large amount, we realize a balanced (non-dominating) comparison between these two measurements using Pareto optimality to assess the qualities of the chosen symbols. This process is shown in Figure 5.1.

Kulic et al.(Kulic, Takano, and Nakamura, 2008) proposed a method to incrementally add observed actions in a hierarchical tree structure, where leaf nodes represent specific motions and more generalized nodes are located closer to the root. The tree is cut into clusters where each cluster corresponds to each symbol. Our method is distinguished by how we measure the validity of the learned activity grammars to choose a better set of symbols (i.e. the feedback). Liang et al.(Liang, Shih, Shih, Liao, and Lin,

2009) trained variable-length Markov models (VLMM) which can automatically learn the model parameters of primitive human actions, where each VLMM is trained to learn each unlabeled action symbol. In our case, however, we do not assume how many action symbols are available. Whereas the codebook and topic models are sequentially obtained for learning action categories in (Niebles, Wang, and Fei-Fei, 2008; Wong, Kim, and Cipolla, 2007), action symbols and activity grammars are found with the feedback in this paper.

As we are concerned with learning syntactic-level action symbols, we preprocess input video sequences into a series of vector representations using low-level feature descriptors. The choice of a low-level descriptor depends on the problem domain, e.g. joint-space description for human motion capture data. Each vector is defined as a group of consecutive frames which share the similar low-level descriptions within the group. They can be regarded as unlabeled video segments.

5.2.1. Discovery of Candidate Symbols

We begin our method by clustering segment vectors using hierarchical agglomerative clustering which incrementally builds a binary tree by grouping a pair of similar vectors based on some distance function, starting from leaf nodes (single-vector nodes). The height of a node represents a distance between two child nodes. By grouping nodes with height less than τ, we obtain κ clusters of vectors. We set initial τ_κ

$$\tau_\kappa = max(\chi(i,j)) \quad \forall i,j \tag{5.1}$$

where inconsistency coefficient $\chi(i,j)$ measures how objects contained in child nodes i and j differ from each other:

$$\chi(i,j) = \frac{d(i,j) - \mu_{i,j}}{\sigma_{i,j}} \tag{5.2}$$

with $\mu_{i,j}$ and $\sigma_{i,j}$ respectively representing mean and standard deviation of heights of all subnodes of i and j.

$$d(i,j) = \sqrt{\frac{2n_i n_j}{n_i + n_j}} \|\bar{x}_i - \bar{x}_j\|_2 \tag{5.3}$$

is a distance function defined using Ward's method to take into account the cost of merging two clusters. Intuitively, the higher the value of $\chi(i,j)$, the less similar the objects belong to that link, hence inconsistent.

The mean of each cluster is used as a symbol description that can classify input video segments and label with its symbol index. We represent a system having κ symbols as ψ_κ. However, as we do not have prior knowledge about whether using κ symbols is optimal to represent an activity effectively, different number of symbols need to be tested: $\Psi = \{\psi_1, \psi_2, ..., \psi_\kappa\}$.

An advantage of using hierarchical clustering analysis is that it does not depend on initial conditions like k-means and provide an intuitive way to partition data points into a desired number of clusters.

5.2.2. Selecting the Number of Symbols

For each system $\psi_\kappa \in \Psi$ obtained in the last section, we build an activity representation from data using acquired symbols. Since training part is unsupervised, we follow the minimum description length (MDL) principles. We require that our training method is able to 1) obtain model parameters in unsupervised way, 2) measure model complexity and likelihood at any

stage of training, and 3) deal with recursions.

Minimum Description Length

As described in Chapter 4, we iteratively apply two types of operators, *Substitute* and *Merge*, until the best grammar is found based on MDL principle. The objective is to find a representation that is sufficiently simple yet expressive, based on the findings reported in Chapter 4 where lower MDL scores generally lead to a better representation. By measuring prior probability of a model $P(M)$ and data likelihood $P(D|M)$, our goal is to minimize the MDL score, represented as $-log$ of joint probability $P(M, D)$:

$$-logP(M, D) = -logP(M) - logP(D|M) \tag{5.4}$$

$$P(M) = P(M_S, M_\theta) = P(M_S)P(M_\theta|M_S) \tag{5.5}$$

where $P(M_S)$ denotes structure prior and $P(M_\theta)$ denotes parameter prior, computed in the same way as in Chapter 4.

Balanced Comparison of Model Complexity and Likelihood

We now train $\psi_\kappa \in \Psi \ \forall\kappa$, i.e. train each system having a different number of symbols. Our goal is to select a system that can describe data well while having reasonable amount of complexity. However, in practice, a model with the lowest MDL score does not guarantee to be the best, as we need exhaustive dataset to compute ideal $P(M)$ and $P(D|M)$. Hence, there is often discrepancy between the value ranges of $-logP(M)$ and $-logP(D|M)$.

Generally, the model description length $-logP(M)$ changes in much higher amount than $-logP(D|M)$ if sampled data were obtained from the same domain, which makes $-logP(M)$ "dominate" MDL score (Kitani, Yoichi,

and Sugimoto, 2008). Hence, it is a common practice to adjust both terms of MDL by multiplying weights to eliminate the biasing problem, but the result still relies on the weights. However, although we do not know the value ranges of the two MDL terms, for sure if both $-logP(M)$ and $-logP(D|M)$ are less than that of another model, it is a better model. This shares the same underlying objective with Pareto optimality.

From this observation, we present a balanced comparison method. First, while performing SCFG learning algorithm which searches for the best model of a system ψ by incrementally changing model parameters, save a pair of MDL components $p = [-logP(M), -logP(D|M)]$ at each step. We obtain these values from all systems $1...\kappa$ and call this set $S = \{p_1, p_2, ...p_n\}$. Compute S^*:

$$S^* = S - \Phi(p_i, p_j) \quad \forall i, j \tag{5.6}$$

where

$$\Phi(p_i, p_j) = \begin{cases} p_i & \text{if } p_i \succ p_j \\ \phi & \text{otherwise} \end{cases} \tag{5.7}$$

and $p_i \succ p_j$ is true only if both components of p_i are larger than p_j, respectively. We vote on S^* how many points belong to each model ψ_κ and choose N-best models. We have now obtained a candidate of models that can represent an activity effectively.

The following figure (Figure 5.2) summarizes the whole process.

5.3. Experiments

5.3.1. Experiment Design

We set our objective to be imitation learning where a robot observes human demonstrator and follow a sequence of actions. Instead of simply imitat-

1. Run unsupervised clustering of unlabeled input signals.
2. Compute candidate systems each having a different set of symbols as described in Section 5.2.1.
3. Apply the grammar learning method described in Section 4.2 on each candidate system and record the MDL scores for every node while constructing the search tree.
4. Perform balanced comparison of model complexity and likelihood for each candidate system using the method described in Section 5.2.2
5. Rank systems based on the number of votes received by each system.

Figure 5.2.: Summary of action symbol selection.

ing, it is required that the system should deal with observation error using the obtained grammars so that it can correctly perform the intended action sequence. Furthermore, similar to the previous experiments, the task representation includes recursion which is demonstrated in various lengths of action sequences, resulting in a more challenging setting.

The experiments are conducted on two different types of dataset. The first one is the Towers of Hanoi dataset used in Chapter 4, on which a visual tracker is applied so that it tracks the current moving block. A single video segment (input to the system) is represented as a 10 dimensional histogram vector computed from the block's quantized positions as well as frame differences (velocities) dx and dy. The second dataset is a newly acquired motion capture Dance dataset, which includes 6 action symbols in a single task grammar. The reason why we had to create an extra dataset is because although the Dance dataset used in the Chapter 4 has non-trivial task structures and contains 9 action symbols in total, only up to 4 action symbols are used in any single task. Similar to (Zhou, Torre, and Hodgins, 2008), 6 most informative joints are selected for learning which makes our

segments to be 6 dimensional vectors. [1]

5.3.2. Findings

We first analyze the Towers of Hanoi dataset. The optimal solution to solve the puzzle requires 5 symbols, which respectively represent a disk to be lifted, placed, and moved between two out of three towers (3 sub-action symbols in total). Figure 5.3 shows an example tree constructed and symbol representations with $\kappa = 8$.

Figure 5.4 shows the spanning values obtained while inducing a grammar for each system ψ_κ. As can be seen, the likelihood does not improve as the number of symbols increases, because the learned model often fails to capture the regularity due to excessive number of symbols. The voting scores in Figure 5.5 suggest that systems ψ_3 and ψ_5 are selected as the best.

This is reasonable since the Towers of Hanoi puzzle can be also represented using 3 symbols, in which case they are interpreted as: "Disk lifted", "Disk dropped", "Disk transferred". However, this is not sufficient to actually solve the puzzle, as the symbol "Disk transferred" is ambiguous, i.e. it only describes *any* movement between two towers. Its representation is actually an averaged histogram of 3 different block transfer actions between two towers, which lacks specificity for execution. This is why systems having 5 symbols failed completely. Our method explicitly takes into account the problem of defining the right "scale" (scope) of a single action, which is generally problem-dependent.

To validate, we parse the input data using the obtained grammar of each system and execute to reproduce actions. During execution, each parsed symbol is mapped to the closest executable action, i.e. one of the five

[1] The experiments in this chapter appear in (Lee, Kim, and Demiris, 2012a).

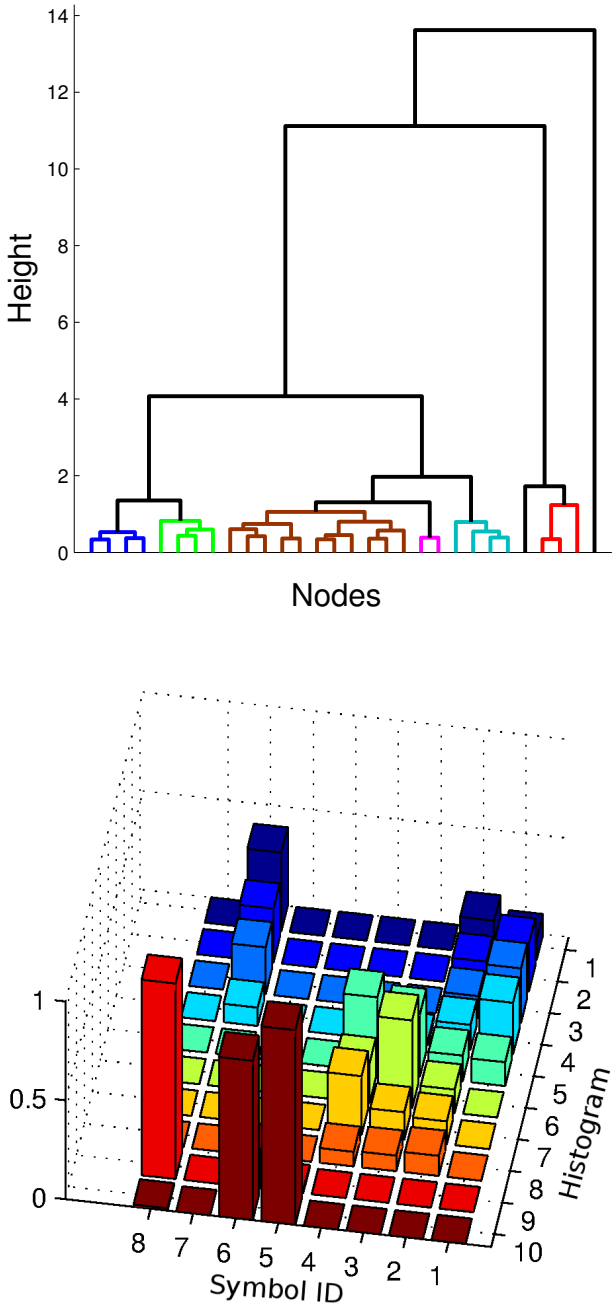


Figure 5.3.: An example clustering tree created (top), showing only the top 30 nodes for better view, and eight action symbol representations (bottom) obtained from the Towers of Hanoi dataset.

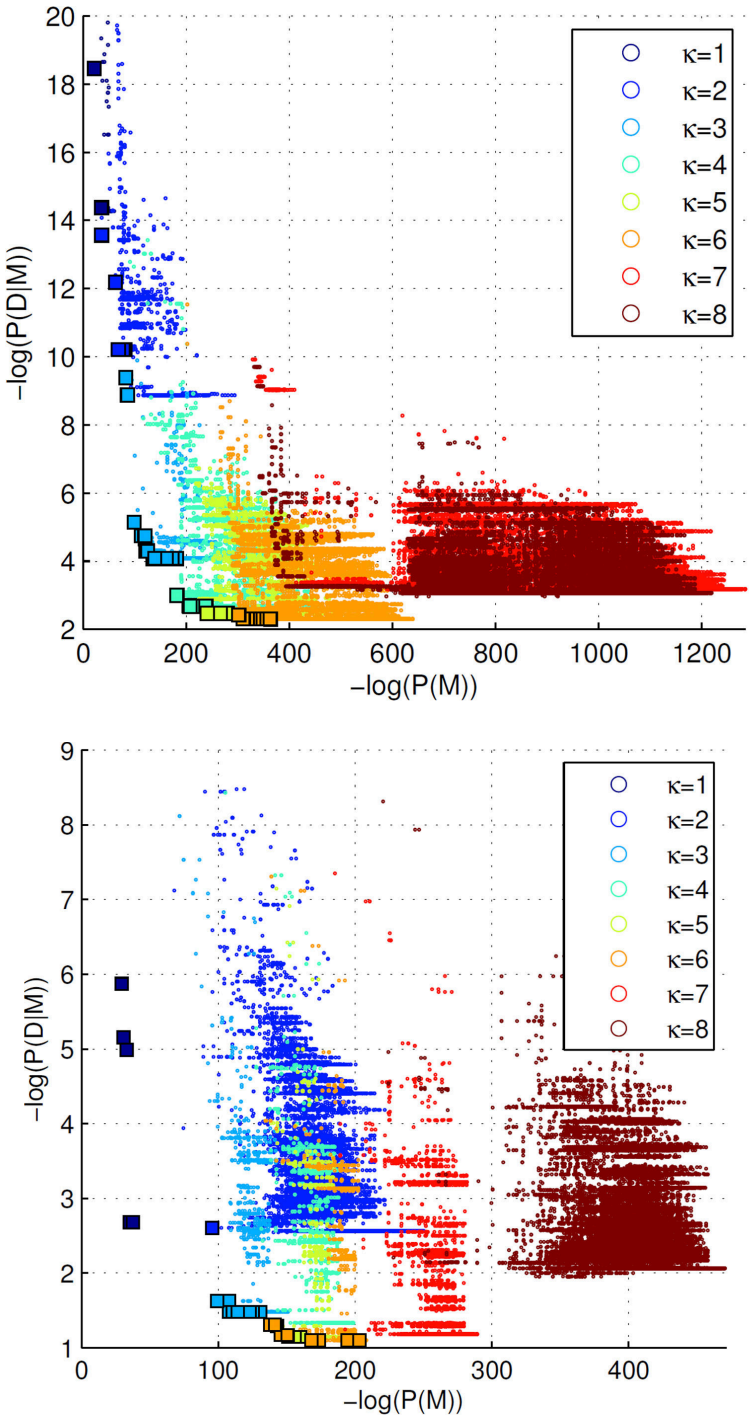


Figure 5.4.: The spanning values of description lengths obtained from the Towers of Hanoi (top) and Dance (bottom) data. Best cases (S^*) obtained using the method described in Sec. 5.2.2 are indicated by square markers. (Best viewed in color.)

κ	α_T	β_T	V_T	S_T	α_C	β_C	V_C	S_C
1	45.3±7.5	16.3±2.0	2	0.00	34.5±4.1	4.0±1.5	5	0.00
2	142.0±42.4	10.1±1.8	6	0.00	177.4±20.4	3.3±0.9	1	0.00
3	202.5±30.8	4.2±0.6	**15**	0.00	124.6±10.9	2.9±1.1	**10**	0.00
4	319.3±43.1	3.0±0.9	8	0.00	173.7±12.0	2.5±1.0	0	0.00
5	356.6±38.2	2.8±0.7	**13**	**0.92**	172.4±8.1	2.5±1.0	4	0.00
6	463.0±58.2	2.5±0.6	9	0.50	191.1±10.7	2.2±1.1	**8**	**0.95**
7	925.3±133.7	3.3±0.4	0	0.92	259.1±13.5	2.0±0.8	0	0.95
8	947.4±114.6	3.1±0.3	0	0.67	413.2±20.8	2.3±0.5	0	1.00

Figure 5.5.: Results on the *Towers of Hanoi* (**T**) and *Dance* (**C**) dataset. α and β denote mean ± standard deviation of $-logP(M)$ and $-logP(D|M)$, respectively. Votes (**V**) are computed by the method described in Section 5.2.2, whereas success rates (**S**) are computed by comparing the parsed symbols.

possible movements mentioned above. As the rule of the puzzle enforces that only a smaller disk shall be placed on top of a bigger disk, there is always only a single possibility of moving a disk between two towers. This is a fair assumption as this rule is always given in prior, not something to be learned. It is marked as success only if the parsed symbols lead to solve the puzzle. ψ_5 showed to be the best considering both success rate and the number of votes, which coincides with the ideal number of symbols.

The Dance dataset is composed of 6 motion primitives (a-f): Raise right or left arm (a, b), Raise both arms (c), Lift left or right leg while raising left or right arm, respectively (d, e), Spin 360° (f). The sample movements are visualized in Figure 5.6. Dance movements are represented as $(abc)^n(def)^n$, where $n = \{1, 2, 3\}$ in our dataset. The result is shown in Figure 5.4. The execution is marked as success only if the parsed symbols exactly match the performed motion primitives.

The sample grammars learned from the Dance dataset are shown in Figure 5.7. As stated above, it was originally demonstrated using 6 symbols. Figure

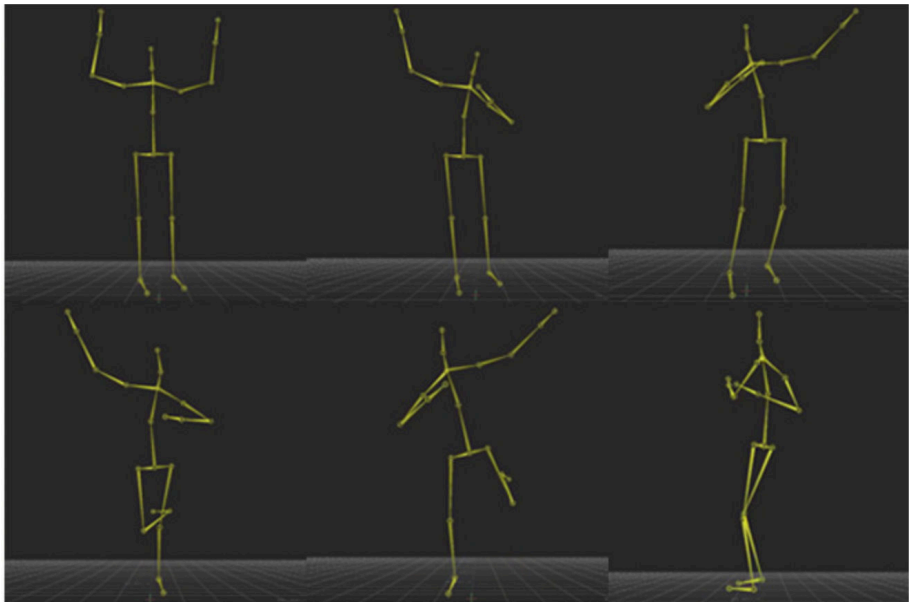


Figure 5.6.: Representative visualized snapshots of the Dance dataset.

S→SEAB [0.399853]
| CFD [0.372613]
| SSEAB [0.199826]
| CEA [0.027708]

(a) 6 symbols (ideal)

S→CDD [0.416571]
| AEBS [0.389128]
| AEBSS [0.179596]
| ACBACDSS [0.014705]

(b) 5 symbols

S→SS [0.347360]
| BCD [0.294369]
| EGF [0.263985]
| SSSS [0.063192]
| EAC [0.020580]
| SEAFSS [0.010513]

(c) 7 symbols

Figure 5.7.: Example grammars learned from data. (a) A grammar generated by a system ψ_6 having 6 symbols A-F. (b) has 1 less symbol, where one of the symbols represents two different actions. (c) has 1 more symbol, where the same action could be represented with two different symbols. Low-probability rules ($< 3\%$) exist due to input data noise.

5.7(a) shows the learned grammar with the ideal number of symbols, which are internally represented as A-F. Figure 5.7(b) shows the case where the system lacks one symbol. As a result, the algorithm needs to reuse one of the symbols to represent 2 actions which are the most similar to each other relative to other actions. In contrast, Figure 5.7(c) shows a grammar represented with 7 symbols, where two symbols could be used to execute the same action. Due to the noise inherent in captured data, there are some erroneous rules having less than 3% rule probabilities.

Note that the results in Figure 5.4 are computed without any knowledge about the success condition, i.e. success rates are used only to verify the validity of the voting results.

5.4. Summary

In this chapter, unsupervised method of selecting models with the "right" number of action symbols was presented. A hierarchical agglomerative clustering analysis and Pareto-inspired voting principles were used to tackle the balancing problem that commonly occurs in MDL score computations. It takes into account the question of choosing the right scope (or resolution) of a single action, which is generally problem-dependent.

Our method exploits the outcomes of SCFG learning technique as feedback to tune the number of symbols, where both grammar learning and symbol discovery are done in unsupervised way. The results confirm that our method is capable to discover and learn the optimal set of action symbols correctly.

The result of the Towers of Hanoi shows an interesting aspect where the proposed method captured the 2 most reasonable models, ψ_5(ideal) and

ψ_3, with notable distinction compared to others. Similarly, in the Dance dataset, ψ_6(ideal) and ψ_3 were chosen, which are also reasonable candidates. The results were obtained without any prior knowledge about the success criteria.

6. Action Anticipation and Attention Allocation using Task Structures

6.1. Introduction

The ability to detect and recognize temporally extended structured activities in crowded dynamic environments is crucial for humans. By considering limited computational resources, which is often neglected but critical in robotics domain, we present a method to actively decide not only "where", but also "when" to retrieve information to maximally improve the overall recognition of task-relevant activities from the scene by exploiting sequential knowledge to optimize the costly sampling of high-dimensional sensory input.

We tackle this problem by taking an information-theoretic approach by integrating the exploitation of the known structures of temporal events in a given domain. For each time step, the motion uncertainties acquired at the low level decides the pan, tilt, and zoom parameters of a camera, where the event uncertainties acquired at the high level decides how much the system has to assign resources in the current view. Our problem differs from conventional event detection problems since our input depends on the

camera parameter chosen in the last time step.

Exploiting the temporal structure of known tasks to allocate attention can allow the perception of several simultaneous events even with sensors having limited field of view. By focusing only on the most discriminative parts of an event, the event could be recognized without complete observation, as the *context* of actions plays an important role on making the system robust to missing observations. For such systems, the ability to predict a good attention timing is crucial to achieving high recognition performances.

In our system, the low-level attentional mechanism uses a low-resolution whole view imagery, similar to retina periphery, to track candidate objects (e.g. people) that may perform activities of interest. The system proposes for each candidate area optimal zoom parameters, aiming to reach higher resolution while minimizing position uncertainty. The functionalities of this component are part of those attributed to the dorsal pathway of human visual perception system (Milner, Goodale, and Vingrys, 2006) which has a major role both in spatial perception and in top-down attention control (Chica, Bartolomeo, and Lupiáñez, 2013).

On the other hand, the high-level attentional mechanism selects one or more areas from the proposed candidates that need to be attended in order to improve the overall recognition rate of the activities in the environment. Intuitively, it is desirable to attend an area that has higher uncertainty about activity hypothesis. Information theoretic approaches are often used to model principled top-down attention mechanisms (Renninger, Coughlan, Verghese, and Malik, 2005; Friston, Adams, Perrinet, and Breakspear, 2012).

This mechanism contains three layers: The lower layer extracts visual features from the attended area, the middle layer recognizes short human actions, and the higher layer integrates previous observations and generates

expectations for future observations. According to (Chinellato and Del Pobil, 2009; Milner, Goodale, and Vingrys, 2006), the low-level visual areas of the brain implements the functionalities of this lower layer. It is also thought that the Action Observation Network (AON) is able to realize the functionalities of the middle layer (Fogassi, Ferrari, Gesierich, Rozzi, Chersi, and Rizzolatti, 2005; Friston, Mattout, and Kilner, 2011; Kilner, 2011; Demiris, 2007). The higher layer is modeled as a mixture of SCFG, which provides a crucial top-down attentional bias based on the internal predictions of the environment changes. This layer may correspond to a ventral pathway, which has been recently proposed (Kilner, 2011) to interact with AON and encode highly abstracted representations of perceived actions.

With this approach, it brings up two interesting policies:

1) Always prefer to watch an area that is most likely to give relevant information.

2) Watch less on highly predictable areas and prefer to search for a new area which is likely to decrease uncertainty the most.

In our experiments, the first strategy corresponds to the minimum entropy attention (MEA) policy, whereas the second strategy corresponds to the maximum mutual information attention (MMIA) policy.

6.2. Using Task Structure Information for Action Anticipation

Our goal is to detect task-relevant activities efficiently by optimally allocating computational resources on potentially multiple regions of interest in a dynamic environment. The benefits of our approach are: 1) It provides a principled method of active camera view selection to maximize the infor-

mation needed to recognize task-relevant activities under resource-bound condition, 2) It allows to detect multiple concurrent activities efficiently by switching among multiple regions of interest, and 3) It allows to detect task-relevant activities as early as possible.

We use an information-theoretic camera control system similar to (Denzler, Zobel, and Niemann, 2003; Sommerlade and Reid, 2008) at the low level, followed by extracting low-level features from the current view to classify human actions. The distribution of human action likelihoods are fed into a high-level task recognition system which is modeled using SCFG. As SCFG is a generative model, it can provide a prediction of future distributions of possible actions given the observations so far, which we exploit to control cameras.

As in the previous chapters, the input to the SCFG parser are the likelihood distribution of primitive action detectors sampled at every time step. Given an observation, the SCFG parser tries to find the best explanation about observations (input stream) which is consistent with the overall expected structure while maintaining temporal consistency. We exploit the result computed during the *prediction step* in Section 2.3.2 to predict the likelihood distribution of actions in the next time step.

Let $\mathbf{E}^k$ be our stochastic variable that describes multiple kinds of activities, which can be one of L activity values, $E_1, E_2, \ldots, E_L$ of object k. Each kind of activity E_l corresponds to a different grammar. An observation o_t^k is a distribution of feature responses computed by action component detectors. Please note that an observation is made only after the camera parameter a_t is selected, i.e. the source of information depends on the camera parameter setting. This is particularly important since several activities evolve simultaneously over time with different speed, the quality of future predictions

that controls the camera parameters, depend on the current decision. For brevity, we will denote observations $o^k_{1,\ldots,t}$ as simply $\tilde{o}^k_t$:

$$P_{a_t}(\mathbf{E}^k = E_j|\tilde{o}^k_t) = P_{a_t}(\mathbf{E}^k = E_j|o^k_{1\ldots t}). \tag{6.1}$$

P_{a_t} denote that all the observations are acquired after selecting the camera parameter a_t. Let $\hat{\mathbf{o}}^k_{t+1}$ be a random variable that denotes the expected observation after the prediction step of parsing. The mutual information between current activity and the observation is:

$$I_{a_t}(\mathbf{E}^k; \hat{\mathbf{o}}^k_{t+1}|\tilde{o}^k_t) = H_{a_t}(\mathbf{E}^k|\tilde{o}^k_t) - H_{a_t}(\mathbf{E}^k|\hat{\mathbf{o}}^k_{t+1}, \tilde{o}^k_t). \tag{6.2}$$

This measurement tells us how much an event uncertainty will change if we make an observation at the next time step, using the expected observation distribution inferred from the high-level knowledge and the observations made so far. $H_{a_t}(\mathbf{E}^k|\tilde{o}^k_t)$, the entropy of event detectors given the observation so far, is computed from the likelihood distributions of action component detectors.

The interesting part for us is $H_{a_t}(\mathbf{E}^k|\hat{\mathbf{o}}^k_{t+1}, \tilde{o}^k_t)$, which requires the expected symbol distribution in the next time step $\hat{\mathbf{o}}^k_{t+1}$.

$$H_{a_t}(\mathbf{E}^k|\hat{\mathbf{o}}^k_{t+1}, \tilde{o}^k_t) = \sum_{\hat{o}^k_{t+1}} P_{a_t}(\hat{\mathbf{o}}^k_{t+1} = \hat{o}^k_{t+1}|\tilde{o}^k_t) H_{a_t}(\mathbf{E}^k|\hat{\mathbf{o}}^k_{t+1} = \hat{o}^k_{t+1}, \tilde{o}^k_t). \tag{6.3}$$

$P_{a_t}(\hat{\mathbf{o}}^k_{t+1}|\tilde{o}^k_t)$ and $P_{a_t}(\mathbf{E}^k|\hat{\mathbf{o}}^k_{t+1}, \tilde{o}^k_t)$ can be obtained from the internal states

of the parser at the *prediction step* (Section 2.3.2):

$$\begin{cases} i : X_k \rightarrow \lambda.Y\mu[\alpha, \gamma] \\ Y \rightarrow \mu \end{cases} \Rightarrow i : Y_i \rightarrow .\nu[\alpha\prime, \gamma\prime] \tag{6.4}$$

$$\alpha\prime = \sum_{\forall \lambda, \mu} \alpha(i : X_k \rightarrow \lambda.Y\mu) P(Y \rightarrow \nu), \gamma\prime = P(Y \rightarrow \nu) \tag{6.5}$$

where $\Rightarrow$ denotes a transition between parser states when the grammar rule $Y \rightarrow \mu$ is applied. α is a *forward probability* that represents the probability of the parsed terminal symbols until i-th index in the input stream, whereas γ is the *inner probability* of substring that starts at input index k and ends at i. ν denotes the possible continuation of input symbols at the current parsing step, i.e. expected observation in the next time step, which is denoted by a '.' notation.

Let s^i be the first symbol of ν in the i-th hypothesis (Equation 6.4) and $\alpha\prime(s^i)$ the forward probability in the i-th hypothesis. Then the observation probability of the q-th action component at time t+1 is:

$$P(\hat{\mathbf{o}}^k_{t+1} = q | \tilde{o}_t) = \sum_{s \in S} \alpha\prime(s^i = q)/\sigma \tag{6.6}$$

where σ is the normalizing factor. This is the expected observation in the next time step given the observations so far.

Now $P_{a_t}(\mathbf{E}^k | \hat{\mathbf{o}}^k_{t+1}, \tilde{o}^k_t)$ can be obtained by simulating the parser to have an input the expected observation $\hat{\mathbf{o}}^k_{t+1}$ and computing the maximum forward probability over all the activity hypotheses at the *prediction* step.

Information-Theoretic Attention Policies

At every time step, the system selects which object w to attend. This corresponds to select the camera parameters among the set $\mathcal{W}_t = \{{a^1}^*_t, {a^2}^*_t, \ldots, {a^N}^*_t\}$.

$\mathbf{E}^w$ will be updated by advancing the parser with the observation received at every time step. For all other objects that were not attended, observation is given as a uniform distribution since there is no new information. This "dummy" observation can be understood as a "missing" data from the parser's point of view. As a result, object k that was not observed maintain the same activity distribution of the previous time step:

$$P_{a^{i*}_t}(\mathbf{E}^k|\hat{\mathbf{o}}^k_{t+1}, \tilde{o}^k_t, a_t = w_i) = P_{a^{i*}_t}(\mathbf{E}^k|\tilde{o}^k_t) \ \forall i \neq k \tag{6.7}$$

We now discuss about how to select the object to attend exploiting the high-level activity knowledge, i.e. the temporal change of observations, encoded in the grammars. A straightforward object selection policy could be to always select the object with the minimum expected entropy at time $t+1$:

$$w^{MEA}_t = \arg\min_k H_{a^{k*}_t}(\mathbf{E}^k|\hat{\mathbf{o}}^k_{t+1}, \tilde{o}^k_t). \tag{6.8}$$

We will name this Minimum Entropy Attention (MEA) policy. This approach drives the system to always follow an object that is most likely to have a known activity. Once the current object activity turns out to be reliable, the system will keep focusing on the current object. However, in scenarios where there are more than one object performing an activity, the system will fail to detect the activities of other objects.

From this motivation, we introduce a more active policy that addresses this problem. Instead of following the minimum entropy object, we try to

minimize the overall expected entropy across all existing object in the scene. We will name this Maximum Mutual Information Attention (MMIA) policy. Thus, for each object k at time t, we define a score function $\mathcal{S}$ as:

$$\mathcal{S}(t,k) = \sum_{j}^{N} H_{a^{k*}_t}(\mathbf{E}^j | \hat{\mathbf{o}}^j_{t+1}, \tilde{o}^j_t). \tag{6.9}$$

The optimal selection policy after making an observation at time t is thus:

$$w_t = \arg\min_{a_t} \mathcal{S}(t, a_t). \tag{6.10}$$

Using Equation 6.7, the difference of $\mathcal{S}$ between any two selections k and j can be reduced to:

$$\mathcal{S}(t,k) - \mathcal{S}(t,j) = I_{a^{k*}_t}(\mathbf{E}^k; \hat{\mathbf{o}}^k_{t+1} | \tilde{o}^k_t) - I_{a^{j*}_t}(\mathbf{E}^j; \hat{\mathbf{o}}^j_{t+1} | \tilde{o}^j_t). \tag{6.11}$$

Thus, minimizing the overall expected entropy is equivalent to attending the object w with the maximum mutual information between $\mathbf{E}^k$ and $\hat{\mathbf{o}}^k_{t+1}$:

$$w^{MMIA}_t = \arg\min_k \mathcal{S}(t,k) = \arg\max_k I_{a^{k*}_t}(\mathbf{E}^k; \hat{\mathbf{o}}^k_{t+1} | \tilde{o}^k_t). \tag{6.12}$$

6.3. Experiments

6.3.1. Experiment Design

In our evaluation, we focus on how our attention allocations are performed on multiple tracks. We assume that we use electronic Pan-Tilt-Zoom (ePTZ) cameras which can transmit both the less detailed images of the whole field of view, as well as more detailed images of a cropped small region. ePTZ cameras, which are commercially available, have advantages over conven-

tional PTZ cameras as they do not need to physically move the camera position which can cause a delay and increase the complexity while tracking objects.

6.3.2. Findings

Synthetic Dataset

We first evaluate our method on synthetic dataset, which is designed to provide an in-depth understanding of how our system behaves under different attention policies. Consider a scenario where objects change colors in some sequence. The temporal structure how colors can change is defined using an SCFG. Our observations are colors of an object in continuous RGB color space. We have two objects in the scene, and our virtual camera system has to choose between two windows to achieve activity recognition performance as close as when watched both.

In this scenario, two objects are involved in two different activities respectively. We define two simple but ambiguous grammars $G_{(R-G-B)^*}$ and $G_{(R-G)^*}$:

$$
\begin{array}{llll}
G_{(R-G-B)^*}: & & & \\
S & \rightarrow & R\ G\ B & [0.5] \\
 & | & S\ S & [0.5]
\end{array}
\qquad
\begin{array}{llll}
G_{(R-G)^*}: & & & \\
S & \rightarrow & R\ G & [0.5] \\
 & | & S\ S & [0.5]
\end{array}
$$

The grammar expression above only shows only the temporal structures and rule probabilities with non-terminal symbols. Symbol emission probabilities are defined in a way that each symbol can have recursions and some noise, e.g. $(R \rightarrow R\ R\ [0.70] \mid r\ [0.29] \mid SKIP\ [0.01])$, where r is a terminal symbol and $SKIP$ is defined as a wildcard that can be substituted with any symbol, as similarly used in (Ivanov and Bobick, 2000; Lee, Kim, and Demiris, 2012b). It is often effective on dealing with noisy data. For brevity,

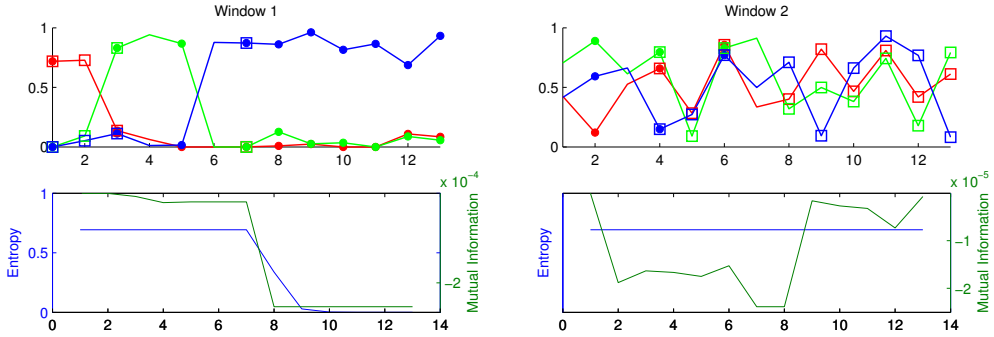


Figure 6.1.: Single vs Random event. Circles and squares denote the attended point when MEA and MMIA were used, respectively. With MEA, window 1 is favored from t=7, whereas with MMIA the system loses interest on window 1 and starts exploring window 2. As a result, window 1 is watched only 4 times, compared to 10 with MEA, without losing too much information that is required to recognize the event happened inside.

we will denote $G_{(R-G-B)^*}$ as G_1 and $G_{(R-G)^*}$ as G_2 in this section.

Single vs Random Event

We show in this section how the systems works when there is no relevant event in the view. The observations in window 2 are generated from uniform distribution, where window 1 contains observations that can be only explained by grammar G_1. We compare the results using the two most meaningful policies, minimum entropy attention (MEA; Equation 6.8) and maximum mutual information attention (MMIA; Equation 6.12) policies.

In Figure 6.1, the entropy values between two windows do not differ much due to the grammar ambiguity until a strong blue color signal is observed. At t=7, the confidence of G_1 increases and as a result, both the entropy and mutual information start dropping. With MEA, window 1 is favored from t=7, whereas with MMIA the system loses interest on window 1 and starts exploring window 2, expecting an event in the future. As a result, window 1 is watched only 4 times with MMIA, compared to 10 with MEA, without

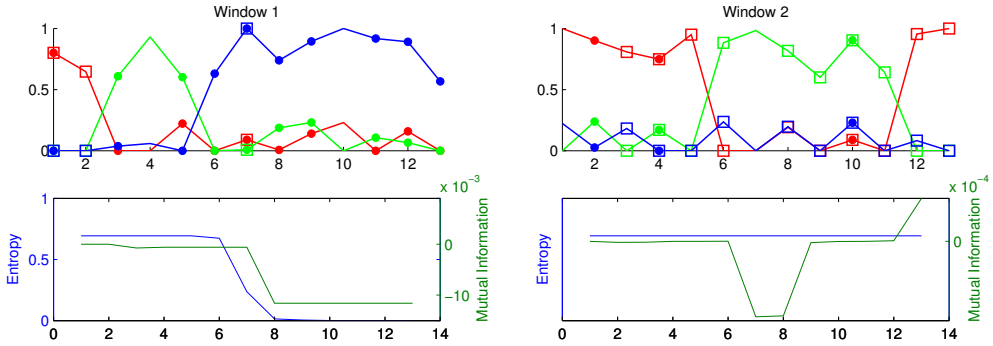


Figure 6.2.: Two concurrent events. Circles and squares denote the attended point when MEA and MMIA were used, respectively.

losing much information required to recognize the event.

Concurrent Events

In Figure 6.2, with MEA, the systems gets interested in watching window 1 from t=6, and gets confident from t=7 after observing strong blue signal. However, although the it does a good job on recognizing the event in window 1, the reappearance of the red signal in window 2 is missed, which can be explained with G_2.

In case of MMIA policy, the system gives more attention in window 2 after t=7 as window 2 did not provide enough information to disambiguate. It also "expects" to observe window 1 at t=7 because window 2 showed a strong green signal at t=6 which lowers the uncertainty. The mutual information drops after confirming a strong blue signal at t=7 in window 1. The system as a result focuses more on window 2 which still needs disambiguation between two events, expecting more information.

VIRAT Dataset

We use a high-resolution video dataset, VIRAT [1] (Oh, Hoogs, Perera, Cuntoor, Chen, Lee, Mukherjee, Aggarwal, Lee, Davis, et al., 2011), since it contains multiple long-term structured activities of two types: *Collection* and *Delivery*. It comes with annotation which includes: *Load/Unload* an object (2 actions), *Open/Close* a car trunk (2 actions), *Get in/out* of a car (2 actions). In addition to these actions, we denote all standing/wandering movements between any two actions as "wander" action. Since the annotation is provided only at the action level, we define the temporal range activities that contain the sequence of these actions.

The *Delivery* activity is defined as: *Get out* of a car, *Open* a trunk, *Unload* an object, *Close* the trunk and *Get in* to the car, whereas *Collection* activity is the same as *Delivery* except it has *load* instead of *unload*. The temporal lengths of these activities allow enough time for our attention system to show effect on our visual system over long time period. It is important to note that an activity may not be carried out by a single person, e.g. one person unloading while another closing the trunk, with abundance of occlusions. Since we perform activity recognition based on the individual level, this situation makes the dataset quite challenging.

To detect actions, we extract spatio-temporal feature descriptors (Alexander Klaser and Schmid, 2008) inside object bounding box using the suggested default values by authors, from which object histogram is computed using bag of words. To train action detectors, we compute accumulated histograms with a fixed-length (50 frames) sliding window and train multi-class linear SVMs using (Chang and Lin, 2011) with probabilistic outputs.

[1] http://www.viratdata.org. We use "VIRAT_S_000001.mp4" to "VIRAT_S_000102.mp4" which include 16 activities, 76 actions (excluding wandering), and 152 human objects.

The detected symbols are fed into SCFG parser to focus the attention. We specify the *Collection* grammar as:

$G_{Collection}$:

$$
\begin{array}{rcllll}
S & \rightarrow & BEFORE & LOAD & AFTER & [0.50] \\
 & | & BEFORE & LOAD & & [0.25] \\
 & | & LOAD & AFTER & & [0.25] \\
BEFORE & \rightarrow & GETOUT & OPEN & & [0.50] \\
 & | & GETOUT & & & [0.25] \\
 & | & OPEN & & & [0.25] \\
AFTER & \rightarrow & CLOSE & GETIN & & [0.50] \\
 & | & CLOSE & & & [0.25] \\
 & | & GETIN & & & [0.25]
\end{array}
$$

Each non-terminal is further defined to allow recursions, e.g. ($LOAD$ $\rightarrow$ $LOAD$ $LOAD$ [0.5] | *load* [0.4] | *SKIP* [0.1]), where *SKIP* symbol is explained in Sec. 6.3.2. *Delivery* activity is also similarly defined.

Note that it is possible to train the grammar from the output of detectors (e.g. (Kitani, Yoichi, and Sugimoto, 2008; Lee, Kim, and Demiris, 2012b)), but due to the small number of activities available in dataset (16 activity samples) and huge variance in detector outputs, the grammar we obtained was not useful to show the differences between the attentional mechanisms studied in this paper.

We show an example activity and the development of window scores under MEA and MMIA policies in Figure 6.3. After a person gets out of the car in window #16, the entropy gets lower and MEA focuses more on it, while MMIA gives attention to #22 on frame 15004. The person in #22 performs the critical action *unload*, which is missed by MEA. MMIA then actively focuses to #20 to see if there is anything informative. We observed

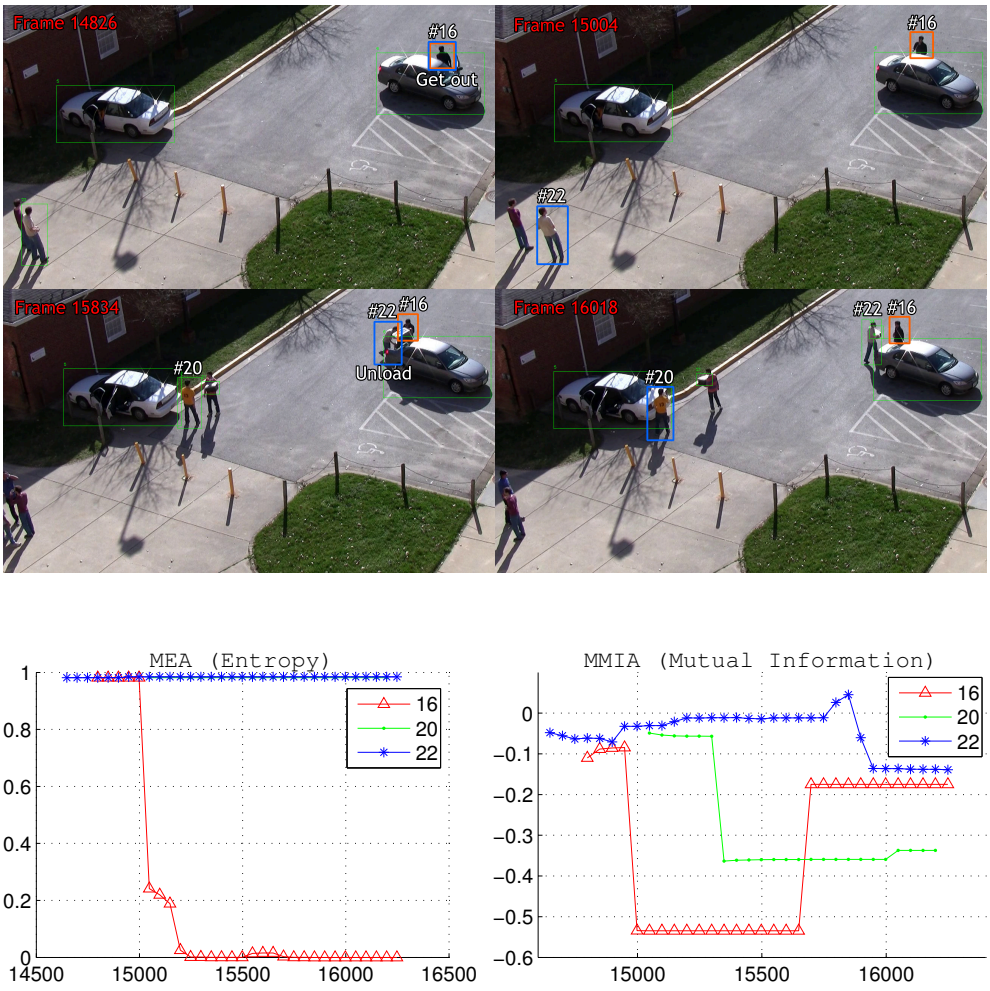


Figure 6.3.: Example *Delivery* task scenario. The blue and red boxes show window attended using MMIA and MEA, respectively. The bottom left and right graphs show the expected entropy of windows under MEA policy, and the mutual information of windows under MMIA policy, respectively. (VIRAT-000006, frames 14648-16277)

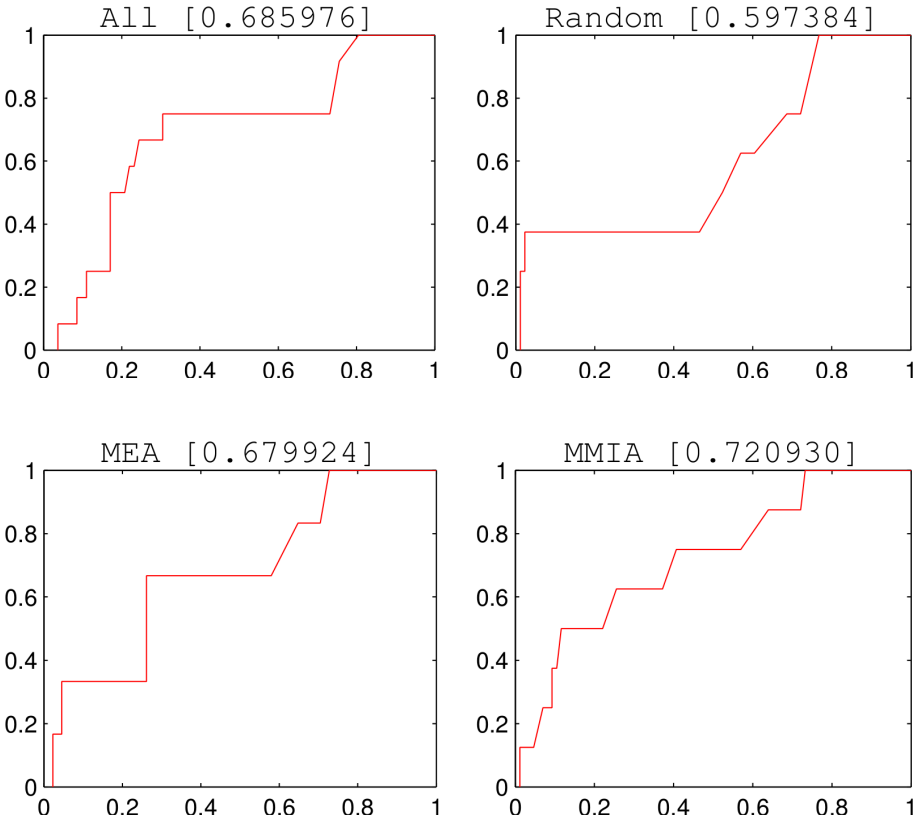


Figure 6.4.: ROC curves obtained under different attention policies and their respective ROC area values.

that under MMIA policy, the system jumps among multiple windows more frequently when compared to MEA.

At the end of every activity, Viterbi parsing is performed to compute the activity likelihoods, normalized by the number of observations. We show ROC curves in Figure 6.4. The reason why MMIA has higher ROC area score than ALL is due to the high level of noise in the data, it sometimes has a side effect of ignoring noise. For ALL policy, incorrectly detected symbols will decrease the overall likelihood since it observes all symbols including wrong symbols. However, as we cannot predict when the noise will happen, this noise filtering effect is not guaranteed.

6.4. Summary

An initial step towards dynamic attention control system for efficient long-term activity recognition has been presented in this chapter. By considering limited computational resources, our method actively decide not only *where*, but also *when* to retrieve information to maximally improve the recognition of temporally structured activities by anticipating the discriminative actions of activities given the limited computational resources.

The structured and abstract representations of activities are crucial for biasing top-down attention to attend the correct object. This bias is measured by considering how the external environment will evolve in the next time step. Our results suggest that taking information-theoretic approach integrated with SCFG formulations show improvement in exploring areas that are likely to provide useful information for recognizing activities.

7. Conclusions and Future Work

7.1. Conclusions

This thesis investigates the problems involved in making a robot to learn and imitate structured human tasks by taking syntactic approaches, adapting the Learning from Demonstrations (LfD) paradigm with emphasis on the symbolic-level task learning and imitation. Through the use of syntactic models such as stochastic context-free grammars (SCFG) for task representation, it is investigated how the knowledge of tasks can be exploited to recognize human behaviors with the purpose of better imitation. This is done through exploiting the semantic constraints of a task, which results in recognizing the human's intention correctly and imitating the correct actions. It results in a more efficient and natural interaction between a human user and a robot by minimizing the need to correct the errors made by the robot while executing actions.

Experimental findings while using SCFG as a task representation framework throughout multiple real-world and simulated tasks are presented including object manipulation games, postural sequence tasks of dance and surveillance tasks. Issues involved in various action detection methods for generating symbols with confidence values on different types of input signals are presented.

This thesis introduces a computational model of structured human task learning that automatically learns task representations from the limited number of human demonstrations containing errors, aiming to discover and extract the important aspects of human tasks in the form of SCFG. It is presented how the learned task representations can be subsequently used to better recognize more complex tasks that share the same underlying action components. It can cope with observation errors as well as human errors that occur both in training and testing stages by explicitly taking into account the uncertainties inherent in action detection. It is experimentally shown throughout various experiments that the quality of the learned task representations under different scenarios coincide with the expected theoretical results. The effect of the grammar rule pruning factor, an important factor while learning SCFG, is experimentally shown and the different results are compared in terms of learning time, model complexity and accuracy.

Taking a step further, an automatic learning method of primitive action symbols are developed, assuming that there is no prior information about what kind of primitive actions are needed to efficiently represent a given task. The question of automatically learning the optimal set of primitive action detectors to describe a task efficiently is investigated and the proposed idea is evaluated throughout the experiments.

Finally, this thesis provides an approach to making use of the task structure information encoded in SCFG with the aim of recognizing task-relevant activities among multiple behaviors observed from people, which results in a step towards dynamic attention control system for efficient long-term task recognition. The structured representations of tasks are exploited to actively decide not only *where*, but also *when* to retrieve information to maximally

improve the recognition of task activities.

7.2. Open Questions and Future Work

One interesting research topic is to augment the parser by adding the "state" information. Currently, the terminals are generated based on events. Hence, it is not suitable to represent simultaneous actions, e.g. *holding an object* ***while*** *approaching a box*. By integrating the notion of *state*, it is possible to describe a wider range of actions more effectively. It is also possible to take advantage of multi-sensory input such as sound or tactile sensing at the same time. This will enable to easily represent the actions between two humans.

Currently, a grammar is parsed independently from other grammars, i.e. a parser does not get affected by the result of another parser. However, in scenarios where a parser's current state in the middle of the observation should affect the state of another parser, it is desirable to have a facility that adjusts the parser's state information based on the state information of other parsers.

Throughout the experiments presented in this thesis, it is assumed that the start and stop point of a task is known. It is possible to alleviate by adding external cues such as vocal commands or gestures made by the demonstrator, although in principle they are essentially equivalent to manual manipulation. This is due to nature of the parser, but it is possible to continuously observe multiple tasks by making the parser to be reset after its likelihood reaches a certain level.

On the execution part, after recognizing the task and parsing the primitive actions, it is possible for a robot to run each action with the same timing

as it had learned from the demonstrator by recording the actual timing between primitive actions. Although execution timing was not critical in the experiments presented in this thesis, one could easily imagine other kinds of tasks where it is more important, e.g. playing musical instruments.

While conducting real-world experiments, the parsing speed has not been negligible when the complexity of the grammar is high and the number of action symbols are large enough (more than 150). This is due to the stochastic nature of the grammar, which generates excessive number of candidate states (hypotheses) during parsing. Balancing between the number of states and accuracy will be a critical component for realizing efficient real-time recognition.

Bibliography

Aggarwal, JK and M.S. Ryoo (2011). "Human activity analysis: A review". In: *ACM Computing Surveys* 43.3, page 16 (cited on page 30).

Ahad, M., JK Tan, HS Kim, and S. Ishikawa (2008). "Human activity recognition: various paradigms". In: *International Conference on Control, Automation and Systems*. IEEE, pages 1896–1901 (cited on page 52).

Alexander Klaser, Marcin Marszalek and Cordelia Schmid (2008). "A Spatio-Temporal Descriptor Based on 3D-Gradients". In: *British Machine Vision Conference*, pages 995–1004 (cited on page 122).

Aloimonos, John, Isaac Weiss, and Amit Bandyopadhyay (1988). "Active Vision". In: *International Journal of Computer Vision* 1.4, pages 333–356 (cited on page 39).

Aloimonos, Y., G. Guerra-Filho, and A. Ogale (2009). "The language of action: a new tool for human-centric interfaces". In: *Human Centric Interfaces for Ambient Intelligence, H. Aghajan, J. Augusto, and R. Delgado (Eds.)* Pages 95–131 (cited on page 32).

Andreopoulos, Alexander and John K Tsotsos (2013). "A Computational Learning Theory of Active Object Recognition Under Uncertainty". In: *International Journal of Computer Vision* 101.1, pages 95–142 (cited on page 40).

Argall, BD, S Chernova, M Veloso, and B Browning (2009). "A survey of robot learning from demonstration". In: *Robotics and Autonomous Systems* 57, pages 469–483 (cited on page 28).

Asada, M., M. Ogino, S. Matsuyama, and J. Ooga (2006). "Imitation learning based on visuo-somatic mapping". In: *Experimental Robotics IX*, pages 269–278 (cited on page 28).

Bajcsy, R. (1988). "Active perception". In: *Proceedings of the IEEE* 76.8, pages 966–1005 (cited on page 39).

Baldwin, DA and JA Baird (2001). "Discerning intentions in dynamic human action". In: *Trends in cognitive sciences* 5.4, pages 171–178 (cited on page 29).

Ballard, D.H. (1991). "Animate Vision". In: *Artificial Intelligence* 48, pages 57–86 (cited on page 39).

Barnachon, Mathieu, Saïda Bouakaz, Boubakeur Boufama, and Erwan Guillou (2014). "Ongoing human action recognition with motion capture". In: *Pattern Recognition* 47.1, pages 238–247 (cited on page 86).

Bentivegna, Darrin C, Christopher G Atkeson, and Gordon Cheng (2006). "Learning similar tasks from observation and practice". In: *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2677–2683 (cited on page 29).

Billard, A. (2001). "Learning motor skills by imitation: a biologically inspired robotic model". In: *Cybernetics and Systems* 32.1, pages 155–193 (cited on page 29).

Billard, A., S. Calinon, R Dillmann, and S. Schaal (2008). *Robot Programming by Demonstration (Chapter 59)*. Springer (cited on page 28).

Billard, Aude, Yann Epars, Sylvain Calinon, Stefan Schaal, and Gordon Cheng (2004). “Discovering optimal imitation strategies”. In: *Robotics and Autonomous Systems* 47.2–3, pages 69 –77 (cited on page 29).

Borji, A. and L. Itti (2013). “State-of-the-Art in Visual Attention Modeling”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 35.1, pages 185–207 (cited on page 39).

Bradski, G. (2000). “The Opencv Library”. In: *Doctor Dobbs Journal* 25.11, pages 120–126 (cited on pages 48, 77).

Breazeal, C. and B. Scassellati (2001). “Challenges in Building Robots That Imitate People”. In: *Imitation in animals and artifacts*, page 363 (cited on page 29).

Breazeal, C. and B. Scassellati (2002). “Robots that imitate humans”. In: *Trends in Cognitive Sciences* 6.11, pages 481–487 (cited on page 29).

Calinon, S., F. Guenter, and A. Billard (2005). “Goal-directed imitation in a humanoid robot”. In: *IEEE International Conference on Robotics and Automation*, pages 299–304 (cited on page 29).

Cangelosi, A. et al. (2010). “Integration of Action and Language Knowledge: A Roadmap for Developmental Robotics”. In: *IEEE Transactions on Autonomous Mental Development* 2.3, pages 167–195 (cited on page 30).

Cangelosi, Angelo and Domenico Parisi (2002). *Simulating the evolution of language*. Springer London (cited on page 30).

Chang, C.C. and C.J. Lin (2011). “LIBSVM: a library for support vector machines”. In: *ACM Transactions on Intelligent Systems and Technology* 2.3, pages 1–27 (cited on pages 88, 122).

Chao, Crystal, Maya Cakmak, and Andrea L Thomaz (2011). “Towards grounding concepts for transfer in goal learning from demonstration”. In: *IEEE International Conference on Development and Learning*. Volume 2, pages 1–6 (cited on page 29).

Chen, D., M. Bilgic, L. Getoor, and D. Jacobs (2011). “Dynamic Processing Allocation in Video”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 33.11, pages 2174–2187 (cited on page 41).

Chica, Ana B., Paolo Bartolomeo, and Juan Lupiáñez (2013). “Two cognitive and neural systems for endogenous and exogenous spatial attention”. In: *Behavioural Brain Research* 237.0, pages 107 –123 (cited on page 112).

Chinellato, Eris and Angel P Del Pobil (2009). “The neuroscience of vision-based grasping: a functional review for computational modeling and bio-inspired robotics”. In: *Journal of Integrative Neuroscience* 8.02, pages 223–254 (cited on page 113).

Chiu, Chih-Yi, Shih-Pin Chao, Ming-Yang Wu, Shi-Nine Yang, and Hsin-Chih Lin (Sept. 2004). “Content-based retrieval for human motion data”. In: *Journal of Visual Communication and Image Representation* 15.3, pages 446–466 (cited on page 88).

Croon, G. de and E.O. Postma (2007). “Sensory-motor Coordination in Object Detection”. In: *IEEE Symposium on Artificial Life*, pages 147–154 (cited on page 39).

Dautenhahn, K and CL Nehaniv (2002). “The Agent-Based Perspective on Imitation”. In: *Imitation in Animals and Artifacts*, pages 1–40 (cited on page 29).

Demiris, J and G Hayes (2002). "Imitation as a Dual-Route Process Featuring Predictive and Learning Components; A Biologically Plausible Computational Model". In: *Imitation in animals and artifacts (Chapter 13), K. Dautenhahn, C. Nehaniv (Eds.)* Pages 327–361 (cited on page 29).

Demiris, Y. (2007). "Prediction of intent in robotics and multi-agent systems". In: *Cognitive Processing* 8.3, pages 151–158 (cited on page 113).

Demiris, Yiannis and Bassam Khadhouri (2006). "Hierarchical attentive multiple models for execution and recognition of actions". In: *Robotics and Autonomous Systems* 54.5, pages 361–369 (cited on page 44).

Denzler, J. and C.M. Brown (2002). "Information theoretic sensor data selection for active object recognition and state estimation". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 24.2, pages 145–157 (cited on pages 39, 40).

Denzler, J., M. Zobel, and H. Niemann (2003). "Information theoretic focal length selection for real-time active 3d object tracking". In: *IEEE International Conference on Computer Vision*. IEEE, pages 400–407 (cited on pages 40, 114).

Dillmann, R (2004). "Teaching and learning of robot tasks via observation of human performance". In: *Robotics and Autonomous Systems* 47.2-3, pages 109–116 (cited on page 28).

Dominey, Peter F (2002). "Conceptual grounding in simulation studies of language acquisition". In: *Evolution of Communication* 4.1, pages 57–85 (cited on page 30).

Dominey, Peter Ford and Jean-David Boucher (2005). "Developmental stages of perception and language acquisition in a perceptually

grounded robot". In: *Cognitive Systems Research* 6.3, pages 243–259 (cited on page 30).

Ekvall, Staffan and Danica Kragic (2008). "Robot learning from demonstration: a task-level planning approach". In: *International Journal of Advanced Robotic Systems* 5.3, pages 223–234 (cited on page 29).

Erlhagen, W, A Mukovskiy, E Bicho, G Panin, C Kiss, A Knoll, H van Schie, and H Bekkering (2006). "Goal-directed imitation for robots: a bio-inspired approach to action understanding and skill learning". In: *Robotics and Autonomous Systems* 54.5, pages 353–360 (cited on page 29).

Fikes, Richard E and Nils J Nilsson (1972). "STRIPS: A new approach to the application of theorem proving to problem solving". In: *Artificial intelligence* 2.3, pages 189–208 (cited on page 96).

Flanagan, JR and RS Johansson (2003). "Action plans used in action observation". In: *Nature* 424.6950, pages 769–771 (cited on page 44).

Fod, A., M.J. Mataric, and O.C. Jenkins (2002). "Automated derivation of primitives for movement classification". In: *Autonomous robots* 12.1, pages 39–54 (cited on page 85).

Fogassi, Leonardo, Pier Francesco Ferrari, Benno Gesierich, Stefano Rozzi, Fabian Chersi, and Giacomo Rizzolatti (2005). "Parietal lobe: from action organization to intention understanding". In: *Science* 308.5722, pages 662–667 (cited on page 113).

Friston, Karl, Rick Adams, Laurent Perrinet, and Michael Breakspear (2012). "Perceptions as hypotheses: saccades as experiments". In: *Frontiers in Psychology* 3.151 (cited on page 112).

Friston, Karl, Jérémie Mattout, and James Kilner (2011). "Action understanding and active inference". In: *Biological cybernetics* 104.1-2, pages 137–160 (cited on page 113).

Gould, Stephen, Joakim Arfvidsson, Adrian Kaehler, Benjamin Sapp, Marius Messner, Gary R Bradski, Paul Baumstarck, Sukwon Chung, and Andrew Y Ng (2007). "Peripheral-Foveal Vision for Real-time Object Recognition and Tracking in Video." In: *International Joint Conference on Artificila Intelligence*. Volume 7, pages 2115–2121 (cited on page 39).

Gurbuz, Sabri, Toshihiro Shimizu, and Gordon Cheng (2005). "Real-time stereo facial feature tracking: mimicking human mouth movement on a humanoid robot head". In: *IEEE-RAS International Conference on Humanoid Robots*, pages 363–368 (cited on page 29).

Higuera, C. de la (2005). "A bibliographical study of grammatical inference". In: *Pattern Recognition* 38.9, pages 1332–1348 (cited on pages 32, 58).

Itti, Laurent and Pierre Baldi (2005). "A principled approach to detecting surprising events in video". In: *IEEE Conference on Computer Vision and Pattern Recognition*. Volume 1. IEEE, pages 631–637 (cited on page 40).

Itti, Laurent, Christof Koch, and Ernst Niebur (1998). "A Model of Saliency-Based Visual Attention for Rapid Scene Analysis". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 20.11, pages 1254–1259 (cited on page 40).

Ivanov, Y. and A. Bobick (2000). "Recognition of visual activities and interactions by stochastic parsing". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 22, pages 852–872 (cited on pages 31, 34, 36, 62, 70, 76, 95, 119).

Jansen, Bart and Tony Belpaeme (2006). “A computational model of intention reading in imitation”. In: *Robotics and Autonomous Systems* 54.5, pages 394–402 (cited on page 29).

Kilner, James M (2011). “More than one pathway to action understanding”. In: *Trends in cognitive sciences* 15.8, pages 352–357 (cited on page 113).

Kitani, K.M., S. Yoichi, and A. Sugimoto (2008). “Recovering the basic structure of human activities from noisy video-based symbol strings”. In: *International Journal of Pattern Recognition and Artificial Intelligence* 22.08, pages 1621–1646 (cited on pages 31, 37, 38, 63, 64, 67, 68, 70, 71, 73, 80, 95, 101, 123).

Kulic, D., W. Takano, and Y. Nakamura (2008). “Combining automated on-line segmentation and incremental clustering for whole body motions”. In: *IEEE International Conference on Robotics and Automation*, pages 2591–2598 (cited on page 98).

Kuniyoshi, Y., M. Inaba, and H. Inoue (1994). “Learning by watching: Extracting reusable task knowledge from visual observation of human performance”. In: *Transactions on Robotics and Automation* 10, pages 799–822 (cited on page 28).

Langley, P. and S. Stromsten (2000). “Learning context-free grammars with a simplicity bias”. In: *The European Conference on Machine Learning*. Volume 1810, pages 220–228 (cited on pages 62, 69).

Larochelle, Hugo and Geoffrey E Hinton (2010). “Learning to combine foveal glimpses with a third-order Boltzmann machine”. In: *Advances in neural information processing systems*, pages 1243–1251 (cited on pages 39, 40).

Lee, Kyu hwa, Jinhan Lee, Andrea Lockerd Thomaz, and Aaron F Bobick (2009). "Effective robot task learning by focusing on task-relevant objects". In: *IEEE/RSJ International Conference on Intelligent Robots and Systems*. St. Louis, USA, pages 2551–2556 (cited on page 29).

Lee, Kyuhwa and Yiannis Demiris (2011). "Towards incremental learning of task-dependent action sequences using probabilistic parsing". In: *IEEE International Conference on Development and Learning*. Volume 2. Frankfurt, Germany, pages 1–6 (cited on pages 31, 45).

Lee, Kyuhwa, Tae Kyun Kim, and Yiannis Demiris (2012a). "Learning Action Symbols for Hierarchical Grammar Induction". In: *The 21st International Conference on Pattern Recognition*. Tsukuba Science City, Japan, pages 3778–3782 (cited on page 104).

Lee, Kyuhwa, Tae Kyun Kim, and Yiannis Demiris (2012b). "Learning Reusable Task Components using Hierarchical Activity Grammars with Uncertainties". In: *IEEE International Conference on Robotics and Automation*. St. Paul, USA, pages 1994–1999 (cited on pages 94, 119, 123).

Lee, Kyuhwa, Yanyu Su, Tae-Kyun Kim, and Yiannis Demiris (2013). "A syntactic approach to robot imitation learning using probabilistic activity grammars". In: *Robotics and Autonomous Systems* 61.12, pages 1323–1334 (cited on page 94).

Liang, Y., S. Shih, A. Shih, H. Liao, and C. Lin (2009). "Learning Atomic Human Actions Using Variable-Length Markov Models". In: *Transactions on System, Man & Cybernetics, Part B* (cited on page 98).

Lockerd, A. and C. Breazeal (2004). "Tutelage and socially guided robot learning". In: *IEEE/RSJ International Conference on Intelligent Robots and Systems*. Volume 4 (cited on page 29).

Lopes, Manuel and José Santos-Victor (2005). "Visual learning by imitation with motor representations". In: *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics* 35.3, pages 438–449 (cited on page 94).

Metta, G., P. Fitzpatrick, and L. Natale (2006). "Yarp: Yet another robot platform". In: *International Journal on Advanced Robotics Systems* 3.1, pages 43–48 (cited on page 148).

Metta, G., G. Sandini, D. Vernon, L. Natale, and F. Nori (2008). "The iCub humanoid robot: an open platform for research in embodied cognition". In: *Proceedings of the 8th Workshop on Performance Metrics for Intelligent Systems*. ACM, pages 50–56 (cited on pages 75, 147).

Milner, A David, Melvyn A Goodale, and Algis J Vingrys (2006). *The visual brain in action*. Volume 2. Oxford University Press Oxford (cited on pages 112, 113).

Moore, D. and I. Essa (2002). "Recognizing multitasked activities from video using stochastic context-free grammar". In: *Proceedings of the National Conference on Artificial Intelligence*. Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999, pages 770–776 (cited on pages 31, 95).

Nevill-Manning, C.G. and I.H. Witten (1997). "Identifying hierarchical structure in sequences: A linear-time algorithm". In: *Journal of Artificial Intelligence Research* 7, pages 67–82 (cited on page 37).

Nevill-Manning, C.G. and I.H. Witten (2002). "On-line and off-line heuristics for inferring hierarchies of repetitions in sequences". In: *Proceedings of the IEEE* 88.11, pages 1745–1755 (cited on page 63).

Nguyen-tuong, Duy and Jan Peters (2008). "Local gaussian process regression for real time online model learning and control". In: *Advances*

in Neural Information Processing Systems, pages 1193–1200 (cited on page 29).

Nicolescu, M.N. and M.J. Mataric (2003). "Natural methods for robot task learning: Instructive demonstrations, generalization and practice". In: *International Joint Conference on Autonomous Agents and Multiagent Systems*, pages 241–248 (cited on page 38).

Niebles, J.C., H. Wang, and L. Fei-Fei (2008). "Unsupervised learning of human action categories using spatial-temporal words". In: *International Joint Conference on Artificila Intelligence* 79.3, pages 299–318 (cited on page 99).

Ogale, Abhijit, Alap Karapurkar, and Yiannis Aloimonos (2007). "View-invariant modeling and recognition of human actions using grammars". In: *Dynamical Vision*, pages 115–126 (cited on page 37).

Ognibene, D. and Y. Demiris (2013). "Towards Active Events Recognition". In: *International Joint Conference on Artificila Intelligence* (cited on pages 40, 41).

Ognibene, Dimitri, G. Pezzulo, and G. Baldassarre (2010). "How can bottom-up information shape learning of top-down attention control skills?" In: *International Conference on Development and Learning*, pages 231 –237 (cited on page 39).

Ognibene, Dimitri, Yan Wu, Kyuhwa Lee, and Yiannis Demiris (2013). "Hierarchies for Embodied Action Perception". English. In: *Computational and Robotic Models of the Hierarchical Organization of Behavior*. Edited by Gianluca Baldassarre and Marco Mirolli. Springer Berlin Heidelberg, pages 81–98 (cited on page 44).

Oh, Sangmin, Anthony Hoogs, Amitha Perera, Naresh Cuntoor, Chia-Chih Chen, Jong Taek Lee, Saurajit Mukherjee, JK Aggarwal,

Hyungtae Lee, Larry Davis, et al. (2011). “A large-scale benchmark dataset for event recognition in surveillance video”. In: *IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, pages 3153–3160 (cited on page 122).

Oliver, Nuria and Eric Horvitz (2005). “Selective perception policies for guiding sensing and computation in multimodal systems: a comparative analysis”. In: *Computer Vision and Image Understanding* 100.1, pages 198–224 (cited on pages 40, 41).

Ota, I., R. Yamamoto, T. Nishimoto, and S. Sagayama (2008). “On-line handwritten Kanji string recognition based on grammar description of character structures”. In: *International Conference on Pattern Recognition*, pages 1–5 (cited on page 31).

Paletta, Lucas, Gerald Fritz, and Christin Seifert (2005). “Cascaded Sequential Attention for Object Recognition with Informative Local Descriptors and Q-learning of Grouping Strategies”. In: *Computer Vision and Pattern Recognition Workshop*. Los Alamitos, CA, USA: IEEE, page 94 (cited on page 39).

Pardowitz, M., S. Knoop, R. Dillmann, and RD Zollner (2007). “Incremental learning of tasks from user demonstrations, past experiences, and vocal comments”. In: *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics* 37.2, pages 322–332 (cited on page 28).

Pastra, K. and Y. Aloimonos (2012). “The minimalist grammar of action”. In: *Philosophical Transactions of the Royal Society B: Biological Sciences* 367.1585, pages 103–117 (cited on page 32).

Pattacini, U., F. Nori, L. Natale, G. Metta, and G. Sandini (2010). “An experimental evaluation of a novel minimum-jerk cartesian controller for

humanoid robots". In: *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1668–1674 (cited on page 77).

Petit, M. et al. (2013). "The Coordinating Role of Language in Real-Time Multimodal Learning of Cooperative Tasks". In: *IEEE Transactions on Autonomous Mental Development* 5.1, pages 3–17 (cited on page 30).

Renninger, Laura Walker, James Coughlan, Preeti Verghese, and Jitendra Malik (2005). "An information maximization model of eye movements". In: *Advances in neural information processing systems* 17, pages 1121–1128 (cited on page 112).

Rizzolatti, Giacomo and Michael A Arbib (1998). "Language within our grasp". In: *Trends in neurosciences* 21.5, pages 188–194 (cited on page 31).

Ryoo, M.S. and JK Aggarwal (2007). "Robust human-computer interaction system guiding a user by providing feedback". In: *International Joint Conferences on Artificial Intelligence*, pages 2850–2855 (cited on page 31).

Sakakibara, Yasubumi, Michael Brown, Richard Hughey, Saira Mian, Kimmen Sjölander, Rebecca C Underwood, and David Haussler (1994). "Recent methods for RNA modeling using stochastic context-free grammars". In: *Combinatorial Pattern Matching*. Springer, pages 289–306 (cited on page 68).

Schaal, S (1999). "Is imitation learning the route to humanoid robots?" In: *Trends in Cognitive Sciences* 3.6, pages 233–242 (cited on page 28).

Shan, Yin, Robert I McKay, Rohan Baxter, Hussein Abbass, Daryl Essam, and HX Nguyen (2004). "Grammar model-based program evolution". In: *Congress on Evolutionary Computation*. Volume 1. IEEE, pages 478–485 (cited on page 68).

Soh, H., Yanyu Su, and Y. Demiris (2012). “Online spatio-temporal Gaussian process experts with application to tactile classification”. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4489–4496 (cited on page 29).

Solan, Z., D. Horn, E. Ruppin, and S. Edelman (2005). “Unsupervised learning of natural languages”. In: *Proceedings of the National Academy of Sciences* 102.33, pages 11629–11634 (cited on page 37).

Sommerlade, Eric and Ian Reid (2008). “Information-theoretic active scene exploration”. In: *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–7 (cited on pages 39, 40, 114).

Sommerlade, Eric and Ian Reid (2010). “Probabilistic surveillance with multiple active cameras”. In: *IEEE International Conference on Robotics and Automation*. IEEE, pages 440–445 (cited on page 40).

Sridharan, Mohan, Jeremy Wyatt, and Richard Dearden (2010). “Planning to see: A hierarchical approach to planning visual actions on a robot using POMDPs”. In: *Artificial Intelligence* 174.11, pages 704–725 (cited on page 39).

Stolcke, A. and S. Omohundro (1994). “Inducing probabilistic grammars by Bayesian model merging”. In: *Grammatical Inference and Applications* 862, pages 106–118 (cited on pages 37, 38, 63–68, 70, 71, 78).

Stolcke, Andreas (1995). “An Efficient Probabilistic Context-Free Parsing Algorithm that Computes Prefix Probabilities”. In: *Computational Linguistics, MIT Press for the Association for Computational Linguistics*. Volume 21 (cited on page 37).

Su, Yanyu, Yan Wu, Kyuhwa Lee, Zhijiang Du, and Yiannis Demiris (2012). “Robust Grasping for an Under-actuated Anthropomorphic

Hand under Object Position Uncertainty". In: *IEEE-RAS International Conference on Humanoid Robots*. Osaka, Japan, pages 719–725 (cited on page 77).

Suzuki, Mototaka and Dario Floreano (2008). "Enactive Robot Vision". In: *Adaptive Behavior - Animals, Animats, Software Agents, Robots, Adaptive Systems* 16.2-3, pages 122–128 (cited on page 39).

Thrun, S. and T.M. Mitchell (1995). "Lifelong robot learning". In: *Robotics and autonomous systems* 15.1-2, pages 25–46 (cited on page 29).

Vijayanarasimhan, S., P. Jain, and K. Grauman (2010). "Far-sighted active learning on a budget for image and video recognition". In: *IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, pages 3035–3042 (cited on page 39).

Viola, Paul and Michael Jones (2001). "Robust Real-time Object Detection". In: *International Journal of Computer Vision* (cited on page 48).

Vogel, J. and N. de Freitas (2008). "Target-directed attention: Sequential decision-making for gaze planning". In: *IEEE International Conference on Robotics and Automation*, pages 2372–2379 (cited on page 39).

Wong, Kwan-Yee Kenneth, Tae-Kyun Kim, and Roberto Cipolla (2007). "Learning motion categories using both semantic and structural information". In: *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–6 (cited on page 99).

Woodward, A.L., J.A. Sommerville, and J.J. Guajardo (2001). "How infants make sense of intentional action". In: *Intentions and intentionality: Foundations of social cognition*, pages 149–169 (cited on page 29).

Wu, Yan and Yiannis Demiris (2010). “Towards One Shot Learning by Imitation for Humanoid Robots”. In: *IEEE International Conference on Robotics and Automation*, pages 2889–2894 (cited on page 29).

Yu, Xiaodong, Cornelia Fermuller, Ching Lik Teo, Yezhou Yang, and Yiannis Aloimonos (2011). “Active scene recognition with vision and language”. In: *IEEE International Conference on Computer Vision*. IEEE, pages 810–817 (cited on pages 40, 41).

Zhou, F., F. Torre, and J.K. Hodgins (2008). “Aligned cluster analysis for temporal segmentation of human motion”. In: *IEEE International Conference on Automatic Face & Gesture Recognition*, pages 1–7 (cited on pages 66, 103).

A. Experimental Setup

A.1. Robot Platform

The experiments are conducted using iCub, an open source 53 degrees-of-freedom (DoF) humanoid robot developed under European Commission's The Seventh Framework Programme. It is approximately 1 meter in height, has two 7-DoF arms, two 9-DoF hands, two 6-DoF legs, a 3-DoF neck, two 3-DoF eyes and a 3-DoF torso. The hand and shoulder joints are tendon-driven, where fingers are pulled against springs with teflon-coated cables running inside teflon-coated tubes. For sensing, it has two cameras on eyes, two microphones on ears, tactile sensors on fingers, distributed capacitive sensors on arms as well as position and torque sensors on major joints (Metta, Sandini, Vernon, Natale, and Nori, 2008). The cameras used are Dragonfly2 firewire cameras developed by Point Grey, from which either 320x240 or 640x480 resolution color images can be retrieved at 30 frames per second. In this thesis, raw images were retrieved in Bayer pattern format and decoded by the camera client module for bandwidth efficiency. Depth calculation was also done based on the images captured from these cameras.

iCub is controlled through either Gigabit Ethernet network or 802.11g/n wireless network through iKart, its supporting platform. The main

software library is YARP (Yet Another Robot Platform), an open source library (Metta, Fitzpatrick, and Natale, 2006).

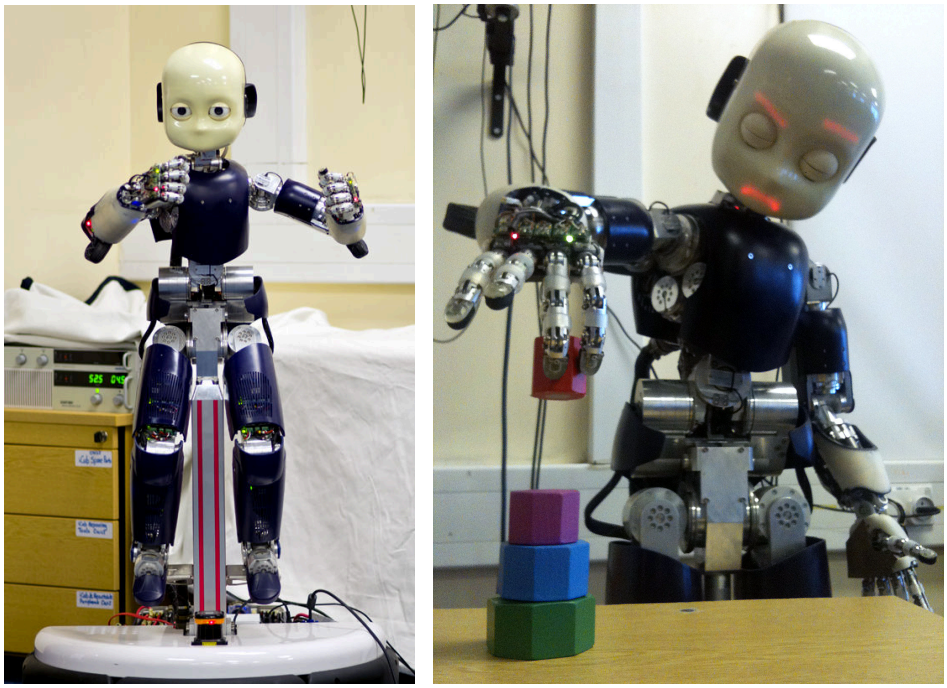


Figure A.1.: iCub performing tasks demonstrated by human partners.

A.2. Motion Capture System

An 8-camera OptiTrack motion capture system developed by Natural Point is used in this thesis for capturing human joint information. It uses passive infrared light reflectance markers attached on a suit. It generates as output 54 dimensional angular values from 18 human joints at 100 frames per second. The processing software is ARENA, also developed by the same company. Using this software, all signals were applied a low-pass filter, automatic gap-filling, marker-joint association and 3D pose computation. Autodesk MotionBuilder software was used for visualization.