Intelligence tests for robots:
Solving perceptual reasoning tasks with a humanoid robot

by

Connor Schenck

A thesis submitted to the graduate faculty
in partial fulfillment of the requirements for the degree of
MASTER OF SCIENCE

Major: Computer Science and Human-Computer Interaction

Program of Study Committee:
Alexander Stoytchev, Major Professor
Vasant Honavar
Jonathan Kelly

Iowa State University
Ames, Iowa
2013

DEDICATION

To my wife, who supported me through the process of writing this thesis.

TABLE OF CONTENTS

LIST OF TABLES vii

LIST OF FIGURES viii

ACKNOWLEDGEMENTS xvi

ABSTRACT xvii

CHAPTER 1. OVERVIEW 1

1.1 Introduction 1

1.2 Research Questions 3

1.3 The Framework 5

1.4 Contributions 6

1.5 Outline 6

CHAPTER 2. RELATED WORK 7

2.1 Intelligence Testing in Psychology 7

2.2 Use of Exploratory Behaviors in Animals, Humans, and Robots 9

2.3 Measuring Object Similarity 13

2.4 Solving Multi-Object Tasks 14

2.4.1 The Odd-One-Out Task 15

2.4.2 The Object Pairing Task and The Montessori Method 18

2.4.3 The Object Ordering Task 20

2.4.4 The Matrix Completion Task 22

2.5 Intelligence Tests in AI 25

2.6 Summary 28

CHAPTER 3. EXPERIMENTAL PLATFORM 30

3.1 The Robot 30

3.2 The Robot's Sensors 32

3.3 The Robot's Behaviors 34

CHAPTER 4. THE OBJECT PAIRING AND MATCHING TASK: TOWARD MONTESSORI TESTS FOR ROBOTS 36

4.1 Introduction 36

4.2 Experimental Setup 38

4.2.1 Robot and Sensors 38

4.2.2 Objects 39

4.2.3 Exploratory Behaviors 40

4.2.4 Data Collection 41

4.3 Feature Extraction 41

4.4 Experimental Methodology 42

4.4.1 Estimating Similarity 42

4.4.2 Combining Sensorimotor Contexts 42

4.4.3 Generating Matchings 45

4.4.4 Evaluation 46

4.5 Results 47

4.5.1 Object Matching with a Single Context 47

4.5.2 Object Matching with Multiple Contexts 47

4.5.3 Repeating the Same Behavior 50

4.6 Summary 51

CHAPTER 5. WHICH OBJECT COMES NEXT? GROUNDED ORDER COMPLETION BY A HUMANOID ROBOT 52

5.1 Introduction 52

5.2 Experimental platform 53

5.3 Feature extraction 55

5.3.1 Sensorimotor feature extraction 55
5.3.2 Object feature extraction 56
5.4 Methodology 56
5.4.1 Problem formulation 56
5.4.2 Selecting the best order completion candidate 57
5.4.3 Order completion using multiple sensorimotor contexts 58
5.4.4 Evaluation 59
5.5 Results 61
5.5.1 An example order completion task 61
5.5.2 Ordering objects using a single sensorimotor context 61
5.5.3 Ordering objects using multiple sensorimotor contexts 62
5.6 Summary 64

CHAPTER 6. WHICH OBJECT FITS BEST? SOLVING MATRIX COMPLETION TASKS WITH A HUMANOID ROBOT 66

6.1 Introduction 66
6.2 Experimental Platform 68
6.2.1 Robot and Sensors 68
6.2.2 Objects 69
6.2.3 Exploratory Behaviors 69
6.2.4 Sensorimotor Contexts 69
6.2.5 Data Collection 72
6.3 Feature Extraction 73
6.3.1 Proprioceptive Feature Extraction 73
6.3.2 Auditory Feature Extraction 73
6.3.3 Visual Feature Extraction 75
6.4 Experimental Methodology 76
6.4.1 Problem Formulation 76
6.4.2 Task Generation 77
6.4.3 Selecting the Best Candidate to Complete a Matrix 79

6.4.4 Extending the Methodology to Columns 81

6.4.5 Measuring Object Similarity 81

6.4.6 Evaluation 85

6.5 Results 86

6.5.1 Performance on a Single Task 86

6.5.2 Performance Across All Tasks 88

6.5.3 Performance Compared to Difficulty of the Task 89

6.6 Summary 93

CHAPTER 7. SUMMARY, CONCLUSION, AND FUTURE WORK 95

7.1 Thesis Summary 95

7.2 Conclusion 96

7.3 Future Work 98

BIBLIOGRAPHY 100

LIST OF TABLES

Table 2.1 Exploratory behaviors identified by Lederman and Klatzky (1987) that are used in deriving particular kinds of knowledge. This table is a close reproduction of the table that appeared in that paper. 11

Table 2.2 Rules identified by previous work and used in the experiments in Chapter 6. 24

Table 6.1 The set of sensorimotor contexts used by the robot. The X's denote modality-behavior combinations that the robot used to solve matrix completion tasks. 72

LIST OF FIGURES

Figure 1.1 The framework used by the robot to solve perceptual reasoning tasks that involve multiple objects. The steps are as follows: 1) The robot explores the objects; 2) It extracts features from the raw sensory data it collected; 3) The robot combines the information from each of the sensorimotor contexts; 4) It computes the value of a task-specific objective function for each candidate solution; and 5) The robot selects the candidate that maximizes the objective function. 5

Figure 2.1 An example odd-one-out task. The correct answer is the red circle because both its color and its shape are different from those of the other three objects. 16

Figure 2.2 An example odd-one-out task posed to the robot by Sinapov and Stoytchev (2010b): a) the similarity matrix (lighter colors indicate higher similarity); b) the Isomap embedding of the matrix shown in a). It is clear from both of these that, in this task, the red hat is the odd object. Reproduced from that paper. 17

Figure 2.3 An example matching task. The objects in each of the two sets vary by size. For each object in the first set (blue objects) there is exactly one object in the second set (red objects) that has the same size. The goal is to find the matching pairs of objects. The correct matching is as follows: (a) goes with (2), (b) goes with (1), (c) goes with (5), (d) goes with (6), (e) goes with (4), and (f) goes with (3). 18

Figure 2.4 A task from the Montessori method. In this task, there are 22 tiles, made up of 11 pairs of the same color. The task is to match the tiles based on their color. 20

Figure 2.5 An example order completion task. The objects in the top row (the blue triangles) are ordered by size, increasing from left to right. The goal is to pick one of the four candidate objects shown in the second row such that it completes this order. The correct answer is (b) because it increases in size in a manner consistent with the first three objects in the ordering. 21

Figure 2.6 An example matrix reasoning task. The correct answer is (c) because it is the same shape as the other objects in the bottom row, it is increasing in size over the previous object in its row, and it completes the permutation over the colors in the bottom row. Looking down the columns, it is the same size and color as the other objects in its column and it completes the permutation over the shapes in the last column. 25

Figure 3.1 The upper-torso humanoid robot. The robot is shown here with some of the Montessori objects that were used in the experiments described in Chapters 4 and 5. 31

Figure 3.2 The Barrett hand attached to the robot's right arm. The image on the left shows the hand with the spread open; the image on the right shows the hand with the spread closed. 31

Figure 3.3 The drive mechanism of the Barrett WAM. The left image shows the cable-and-cylinder drive for one of the joints; the right image shows the layout of the pucks in the WAM. Adapted from Rooks (2006). 32

Figure 3.4 The robot's sensors: a) one of the robot's 2 RGB Logitech Webcams; b) one of the robot's 2 Audio-Technica U853AW microphones. 33

Figure 3.5 The robot's Microsoft Kinect RGBD camera. It was attached to the metal stand supporting the robot during the experiments described in Chapter 6. 34

Figure 3.6 An example data stream recorded by the robot during one interaction with an object (the behavior in this case is *rattle*). The top row of the figure shows a selection of images recorded by the robot's Kinect camera. The middle row shows the audio spectrogram computed from the audio data recorded from the robot's microphones. The bottom row shows the torques applied to the robot's arm during the interaction. . 35

Figure 4.1 The robot and the four Montessori matching tasks that were used in the experiments. In clockwise order, the four tasks were: sound cylinders, weight cylinders, pressure cylinders, and sound boxes. 38

Figure 4.2 The four sets of Montessori objects used in the experiments. From left to right and top to bottom the object sets are: *pressure cylinders*, *sound boxes*, *sound cylinders*, and *weight cylinders*. All objects are marked with colored dots on the bottom to indicate the correct matches; other than that, the objects in each set are all visually identical (except for the *pressure cylinders* and the *sound cylinders*, which also have different colors for the tops to indicate the two sets of six objects). 39

Figure 4.3 The ten exploratory behaviors that the robot performed on all objects. From left to right and top to bottom: *grasp*, *lift*, *hold*, *shake*, *rattle*, *drop*, *tap*, *poke*, *push*, and *press*. The object in this figure is one of the sound boxes. The red marker on the table indicates the initial position of the objects at the beginning of each trial. The object was placed back in that position by the experimenter after some of the behaviors (e.g., drop). 40

Figure 4.4 The similarity matrices used to perform matching given two sets of six objects each for the *sound cylinders*. The matrices for each individual context are shown as well as the consensus matrix for all 20 contexts ("**A**" denotes matrices computed from *audio* and "**P**" denotes matrices computed from *proprioception*). The pairing accuracy combination method using four pairs for training was used to combine the individual matrices. In each matrix lighter colors denote more similarity while darker colors denote more dissimilarity. 43

Figure 4.5 The consensus weight matrix for the *sound cylinders* using all 20 sensorimotor contexts for matching two groups of six objects. The pairing accuracy combination method using four pairs to train was used to combine the individual similarity matrices for each context. The subscripts indicate correct matches. This matrix is identical to the one shown in the right-hand side of Figure 4.4. 44

Figure 4.6 The accuracy of each context when matching between two sets of six objects. Lighter values indicate higher accuracy with completely white being 100%. Darker values indicate lower accuracy with completely black being 0%. The images from left to right are: *pressure cylinders*, *sound boxes*, *sound cylinders*, and *weight cylinders*. 48

Figure 4.7 The kappa statistic for each set of objects. Each line represents a different method for combing the sensorimotor contexts. The line labels are as follows: U-uniform combination; R-recognition accuracy based combination; P2-pairing accuracy using two pairs for training; P3-pairing accuracy using three pairs for training; P4-pairing accuracy using four pairs for training. 49

Figure 4.8 The kappa statistic averaged across all four sets of objects while varying the number of interactions used to generate the similarity matrices $\mathbf{W}^c$ for each context $c \in \mathcal{C}$. The number of randomly sampled interactions was varied from 1 to 9. The line labels are the same as in Figure 4.7. 50

Figure 5.1 The three sets of objects used in the experiments 54

Figure 5.2 The ten exploratory behaviors that the robot performed on the objects. From left to right and top to bottom: *grasp*, *lift*, *hold*, *shake*, *drop*, *tap*, *poke*, *push*, *press*, and *rattle*. The *rattle* behavior wasn't performed on the *cones* and *noodles*. The object in this figure is one of the pressure cylinders. After some of the behaviors (e.g., drop), the object was moved back to the red marker location on the table by the experimenter. . . . 55

Figure 5.3 An example task. The box on the left shows both the ordered set $\mathcal{L}$ and the unordered set of objects $\mathcal{G}$ to choose from. The plot on the right shows the ISOMAP embedding of the distance matrix between the objects. The blue circles denote the three objects in $\mathcal{L}$, the red circle denotes the object in $\mathcal{G}$ that is selected to complete the order. . . 60

Figure 5.4 The accuracy of each context for each of the 3 concepts. Darker values indicate lower accuracies with solid black being 0%; lighter values indicate higher accuracies with solid white being 100%. 62

Figure 5.5 The accuracy as the number of contexts is increased. The blue line is the accuracy when picking the single-best context; the cyan line is the accuracy when using uniform weights to combine contexts; the red line is the accuracy when the contexts are weighted in proportion to their individual accuracies; and the green line is the accuracy achieved when using AdaBoost to learn the weights. 63

Figure 5.6 The average accuracy as the number of tasks used for training is increased. The blue line is the accuracy achieved when picking just the single-best context; the red line is the accuracy achieved when the contexts are weighted in proportion to their individual accuracies; and the green line is the accuracy achieved when using boosting. The results are averaged over 50 sets of training tasks for each size from 1 to 49. . 64

Figure 6.1 The robot used in these experiments. It is shown here with only its right arm as the left arm was temporarily removed for maintenance when these experiments were performed. The Microsoft Kinect camera is mounted on the lower part of the robot's torso. 68

Figure 6.2 The properties by which the objects varied. Each object is a jar that is one of three colors, filled with one of four different types of contents, and weighing one of three different weights, for a total of 36 objects (see Figure 6.3). 70

Figure 6.3 The 36 objects used in the experiments described in this chapter, grouped by color. Within each group, all objects of the same weight are in the same row and all objects with the same type of contents are in the same column. 70

Figure 6.4 Before and after images for the ten exploratory behaviors that the robot performed on all objects. From left to right and top to bottom: *grasp*, *lift*, *hold*, *shake*, *rattle*, *drop*, *tap*, *poke*, *push*, and *press*. The object was placed back in the initial position by the experimenter after some of the behaviors (e.g., drop). 71

Figure 6.5 An example sensory record of proprioceptive values. The top image depicts the raw joint torques recorded from the robot's arm during the interaction where light values denote positive torque values and dark values indicate negative torque values. The bottom image depicts the features extraced from that by binning the values for each of the 7 joints into 10 temporal bins. 74

Figure 6.6 An example sensory record of auditory values. The top image depicts the audio spectrogram, which was computed using a series of DFTs on the raw wave that was recorded during the interaction (red denotes higher activation, green and blue denote lower activation). The bottom image depicts the features extracted from the spectrogram by binning the values into 10 temporal bins and 10 frequency bins. 75

Figure 6.7 An example image recorded during the look behavior. The left image shows the raw RGB data that the robot's camera recorded. The middle image is the segment of the robot's field of view where the object was always placed on the table. The right image is the 8×8 grid that this segment was binned into, where each location in the grid is the average of the pixels that fall into that cell. 76

Figure 6.8 The algorithm that prunes the set of distance functions. It takes as input an initial set of distance functions $\mathcal{D}$ and a set of training tasks $\mathcal{L}$ for which the correct answers are known. The method $evaluatePerformance$ returns the accuracy of the given set of distance functions on the given training tasks. 84

Figure 6.9 An example matrix reasoning task solved by the robot as part of this experiment. The words below each object in the matrix represent the values for each of the three properties for that object. The bars next to each candidate object represent the normalized objective function values for each of the four distance methods. The correct answer is (g). 87

Figure 6.10 Accuracy versus number of contexts used to solve the tasks. As expected, the accuracy improves as the robot is allowed to use information from more sensorimotor contexts. Each line represents a different distance function method. The two category distance methods perform better than the two context distance methods. As expected, the category method with supervision performs the best. 88

Figure 6.11 Accuracy versus number of contents for the subset of tasks that don't require color information and for all tasks. The line labeled "Overall" is the same as the line labeled "Category Distances + Supervision" in Figure 6.10. The other line was computed in the exact same way as the overall line with the exception that the 500 tasks were reduced to just the 173 that did not require perception of color to solve. The standard deviation for each data point is also plotted using dashed lines. 89

Figure 6.12 Six figures that compare the performance as a function of the difficulty of the matrix completion tasks. 90

Figure 6.13 Three figures that show the number of matrix completion tasks for different task difficulty types. 91

ACKNOWLEDGEMENTS

I would like to acknowledge Jivko Sinapov, David Johnston, and Alexander Stoytchev, who were my co-authors on the research papers that lay the foundation for the work done in this thesis. They helped me develop the ideas presented in this thesis as well as collect and process the data and publish the results. Much of the work done in this thesis was inspired by work done by Jivko Sinapov. Additionally, Dr. Alexander Stoytchev served as my major adviser throughout this process and helped me brainstorm ideas, develop methodology, and analyze the results. I would also like to acknowledge Vladimir Sukhoy and Taylor Bergquist for their help with various aspects of this thesis.

I must acknowledge the Developmental Robotics Laboratory, the Department of Computer Science, the Virtual Reality Applications Center, and Iowa State University for all their support. I would also like to acknowledge the National Science Foundation and the Human Computer Interaction Program at ISU for their financial support of the work done in this thesis. This work was partially supported by the National Science Foundation Graduate Research Fellowship Program (NSF Grant No. DGE1247194).

ABSTRACT

Intelligence test scores have long been shown to correlate with a wide variety of other abilities. The goal of this thesis is to enable a robot to solve some of the common tasks from intelligence tests with the intent of improving its performance on other real-world tasks. In other words, the goal of this thesis is to make robots more intelligent. We used an upper-torso humanoid robot to solve three common perceptual reasoning tasks: the object pairing task, the order completion task, and the matrix completion task. Each task consisted of a set of objects arranged in a specific configuration. The robot's job was to select the correct solution from a set of candidate solutions.

To find the solution, the robot first performed a set of stereotyped exploratory behaviors on each object, while recording from its auditory, proprioceptive, and visual sensory modalities. It used this information to compute a set of similarity scores between every pair of objects. Given these similarity scores, the robot was able to deduce patterns in the arrangement of the objects, which enabled it to solve the given task. The robot repeated this process for all the tasks that we presented to it. We found that the robot was able to solve all the different types of tasks with a high degree of accuracy.

There have been previous computational solutions to tasks from intelligence tests, but no solutions thus far have used a robot. This thesis is the first work to attempt to solve tasks from intelligence tests using an embodied approach. We identified a framework for solving perceptual reasoning tasks, and we showed that it can be successfully used to solve a variety of such tasks. Due to the strong correlation between intelligence test scores and performance in real-world environments, this suggests that an embodied approach to learning can be very useful for solving a wide variety of tasks from real-world environments in addition to tasks from intelligence tests.

CHAPTER 1. OVERVIEW

1.1 Introduction

Intelligence tests have been around for over 4000 years (DuBois, 1970). Some ancient Chinese emperors gave proficiency and ability tests to state officials. Even an 8-year-old Wolfgang Amadeus Mozart was given an intelligence test when he appeared before King George III of England in 1763 (Gregson, 1989). Indeed, human societies have been interested in tests of mental ability for a long time. Modern intelligence tests, though, bear little resemblance to their historical predecessors. Modern tests, such as the Wechsler Intelligence Scale for Children (WISC), often measure test takers' abilities in areas such as verbal comprehension, perceptual reasoning, working memory, and processing speed (Wechsler, 2003). Other tests, such as the Raven's Progressive Matrices (RPM) test (Raven, 1938), focus on one specific area (in the case of the RPM, matrix reasoning). All of these tests though, have been shown to correlate with performance on a variety of other mental tasks (Neisser et al., 1996; Hunter and Schmidt, 1998; Deary et al., 2007; Naglieri and Bornstein, 2003).

The goal of this thesis is to investigate the ability of robots to solve tasks that commonly appear on intelligence tests. More specifically, we want to know how a robot can go about solving these kinds of tasks and how that can inform the development of more intelligent robots. Intelligence tests are an interesting research domain for roboticists because they are well-founded in the psychology literature and have been shown to have meaningful correlations with other measures of intelligence. Furthermore, many of the concepts underlying the tasks in intelligence tests also appear frequently in human environments. Thus, by studying how robots can solve intelligence tests, we can better understand how to make robots more intelligent in a manner grounded in our understanding of human intelligence and in a manner well-suited for the environments in which they will operate.

Intelligence tests are often divided into a variety of areas such as verbal comprehension, working memory, and perceptual reasoning. In this thesis, however, we focus exclusively on perceptual reasoning tasks. These tasks are well-suited for robots because they focus on the perception of object properties and reasoning about those properties in order to solve the tasks. Other types of tasks are not as well-suited for robots. Some require large amounts of background knowledge, something that robots are not yet capable of acquiring on their own. For example, verbal comprehension requires that a robot first understand language before it can solve the tasks, something that no robot can currently do. Other tasks are designed to measure individual differences in humans that do not occur in robots. For example, working memory tasks are designed to measure how much information a person can store in memory at one time, but with the large amounts of memory available in modern computers, these tasks are not as useful for testing on robots. Perceptual reasoning tasks, though, do not require large amounts of prerequisite knowledge to solve, and they measure reasoning ability rather than individual differences that occur only in humans.

Fluid reasoning, or the ability to solve novel problems, is very similar to perceptual reasoning and was proposed as part of Horn's theory of intelligence (Horn and Cattell, 1966). One of the best measures of fluid reasoning ability is the RPM, a test composed entirely of matrix reasoning tasks (Kaufman, 2009). The concepts that underlie matrix reasoning tasks appear in many places outside of intelligence tests. For example, Mendeleev created the periodic table as a grid because he noticed that many of the elements' properties fit into a repeating pattern (Scerri, 2011). This arrangement of the elements allowed Mendeleev to successfully predict the existence of many as yet undiscovered elements (Scerri, 2011).

The regularity exhibited by the periodic table prompted other scientists to wonder whether or not atoms really were the most elementary particles, which led to the discovery of subatomic particles (Lincoln, 2012). More recently, the Standard Model of particle physics was developed, which posits that matter is composed of quarks and leptons (Gaillard et al., 1999). When the various types of quarks and leptons are arranged in a grid based on their properties, many regularities between them become apparent. This has prompted some scientists to wonder if this regularity implies that these particles are composed of as yet undiscovered, even more elementary particles (Lincoln, 2012). In both of these cases, the arrangement of objects into a grid structure made obvious otherwise undiscovered patterns among the objects, suggesting that the concepts underlying matrix completion tasks (and perceptual reasoning tasks in general) have many applications outside intelligence tests.

1.2 Research Questions

The main research question addressed in this thesis is:

> **How can a robot solve multi-object perceptual reasoning tasks using embodied representations of the objects?**

In order to answer this question, we looked into three subsidiary questions, listed below.

1. **How can a robot solve the object pairing task?**

The object pairing task is one of the most basic tests of perceptual reasoning ability. The test taker has to find matches between two groups of objects. Often the matchings are based on a single object property, and so the test taker must deduce this property and then simply match the objects that are most similar with respect to this property. The object pairing task also commonly appears as a task in the Montessori method, which is an alternative style of education (Pitamic, 2004). Among other things, the Montessori method focuses on self-directed learning in concert with specialized learning materials (Montessori, 1912; Lillard, 2008; Lillard and Else-Quest, 2006). Children are given sets of objects that are designed to vary by specific properties in order to teach the children about a specific concept (Pitamic, 2004). For example, the sound cylinders are a set of wooden, cylindrical toys filled with different types of contents, where every toy has a matching toy that is filled with the same type of contents. The task is designed to teach children about auditory similarity between visually identical objects and how that relates to pairing them. Since many Montessori objects are designed to vary by specific properties and be identical in all others, they are well-suited for testing a robot's ability to pair objects based on their physical properties. This is explored further in the experiments described in Chapter 4.

2. **How can a robot solve the order completion task?**

Unlike the pairing task, the order completion task not only requires the test taker to deduce the property by which the given ordering varies, but also to continue the ordering in the same direction. This task is common on intelligence tests. For example, in the Intelligence and Development Scales test, children are asked to sort line segments of varying length (Hagmann-von Arx et al., 2008). The order completion task is a good way to evaluate the test taker's ability to understand the basic object properties and how to relate objects based on those properties. Indeed, studies have shown that young children have some of the underlying abilities required to solve this task. For example, children from 2 to 4 years old have been shown to understand "big" and "small" when comparing objects (Graham et al., 1964; Ebeling and Gelman, 1988, 1994). This suggests that evaluating a robot on the order completion task may be a good domain to test if robots can learn to order objects as well. This is explored further in the experiments described in Chapter 5.

3. **How can a robot solve the matrix completion task?**

Whereas the pairing and order completion tasks require the test taker to perceive a *single* property by which the objects vary and then solve the task based on that property, the matrix completion task requires that the test taker be able to perceive and reason about multiple object properties at once in order to solve the tasks. This task involves reasoning about a set of objects placed in a grid in order to complete the grid. Matrix completion tests have been shown to predict performance on a wide-range of other problem solving tasks (Prabhakaran et al., 1997; Fuchs et al., 2008). This suggests that the ability to synthesize different types of information is fundamentally important to being able to solve complex problems. In the experiments described in Chapter 6, we look at how a robot can solve matrix completion tasks.

The complexity of these three tasks gradually increases. To solve the object pairing task, the robot must be able to perceive the property by which the objects vary, then it must simply find the pairs of objects that have the same value for that property. In the second task, the

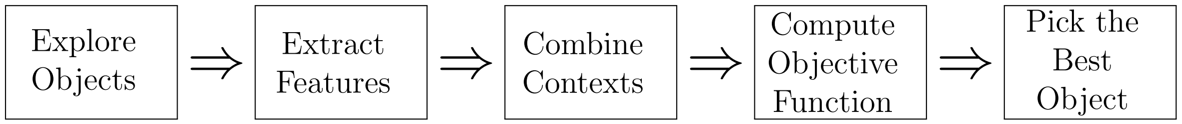


Figure 1.1: The framework used by the robot to solve perceptual reasoning tasks that involve multiple objects. The steps are as follows: 1) The robot explores the objects; 2) It extracts features from the raw sensory data it collected; 3) The robot combines the information from each of the sensorimotor contexts; 4) It computes the value of a task-specific objective function for each candidate solution; and 5) The robot selects the candidate that maximizes the objective function.

order completion task, the robot must not only be able to perceive the property by which the objects vary, but also it must be able to reason about that property in order to complete the ordering. In the matrix completion task, the robot must be able to perceive multiple properties by which the objects vary at once, and it must be able to reason about that variation in order to complete the matrix. Thus, these tasks progressively test the robot's ability to detect object properties, to reason about those properties, and to be able to detect and reason about multiple properties at once.

1.3 The Framework

Figure 1.1 shows the framework that we will use in this thesis to solve mutli-object tasks. It has been developed over the course of previous work in the Developmental Robotics Laboratory. The framework is implemented as follows. Given a task, the robot first explores the objects in the task by performing a set of stereotyped behaviors on them while recording from multiple sensory modalities. Second, the robot extracts features for each object from the sensory records for that object and computes the perceptual similarity between every pair of objects in each sensorimotor context (where a sensorimotor context is defined as the sensory record for one modality during one behavior). Third, the robot combines the information from the different contexts together using one of several methods. Fourth, based on the previous step, the robot computes the value of a task-specific objective function for each possible solution to the given task. Finally, it selects the solution that optimizes the function. The specific implementation of this framework is described in more detail in Chapters 4, 5, and 6.

1.4 Contributions

The main contribution of this thesis is to the understanding of how robots can solve tasks that commonly appear on intelligence tests. This is done primarily through the validation of the computational framework described in the previous section. While this thesis is not the first work to use this framework, it does develop it further and validate it on a variety of tasks. We show that grounded learning based on exploratory behaviors is not only a valid way to learn about objects, but also that it can be used to solve multiple perceptual reasoning tasks. We also show that combing information from heterogeneous sensorimotor contexts can improve the performance of robots on these tasks. Finally, we show that it only takes a minimal amount of supervision in the form of example tasks in order for a robot to learn the important aspects of a given task. All of this together shows not only that robots can solve perceptual reasoning tasks, but also that they can do so based on their own embodied sensorimotor representations with little human supervision.

1.5 Outline

The rest of this thesis is organized as follows. Chapter 2 goes over the related work in psychology, robotics, and AI. Chapter 3 describes the experimental setup and the robot that was used to perform the experiments. Chapter 4 describes the first set of experiments, which addresses the first research question, by showing how a robot can solve the object pairing task. Chapter 5 describes the second set of experiments, which addresses the second research question, by showing how a robot can solve the order completion task. Chapter 6 details the third set of experiments, which extends the previous two chapters, and addresses the third research question by showing how a robot can solve the matrix completion task. Finally, Chapter 7 summarizes the results, draws conclusions based on them, and suggests possible experiments for future work.

CHAPTER 2. RELATED WORK

2.1 Intelligence Testing in Psychology

One of the first theories of intelligence was proposed by Spearman (1904), who posited that "intelligence is largely a single global ability" (Kaufman, 2009). Spearman noticed strong correlations between measures of different mental abilities and proposed the two factor theory of intelligence (Spearman, 1914). He postulated that there must be a general factor correlated with all mental abilities, which he named g, and a specific factor correlated with individual abilities, which he called s. Horn and Cattell (1966) expanded on this theory by breaking g down into fluid intelligence (denoting a person's ability to solve novel problems, referred to as Gf) and crystallized intelligence (denoting the knowledge a person acquires from school and culture, referred to as Gc). More recently their theory has been merged with Carroll's *Three Stratum Theory* of intelligence (Carroll, 1997) to form the Cattell-Horn-Carroll Theory, or CHC theory (Kaufman, 2009). CHC theory breaks intelligence into ten components: fluid intelligence, crystallized intelligence, quantitative reasoning, reading and writing ability, short-term memory, long-term storage and retrieval, visual processing, auditory processing, processing speed, and decision speed/reaction time (Kaufman, 2009). In this thesis we are interested in developing methods to solve problems focusing on fluid reasoning, or the ability to solve novel problems. The experiments in this work focus on understanding how robots can solve tasks that emphasize reasoning over immediately perceivable object properties rather than on knowledge built over a long period of time.

Modern intelligence tests have existed as long as, if not longer than, modern theories of intelligence. The first modern intelligence test was proposed by Sir Francis Galton (Cohen et al., 1999). It evaluated test takers' abilities in areas such as keenness of sight, steadiness of hand, and strength of pull (among others). The French psychologist Alfred Binet dissented from this test believing that "the concept of measuring something so complex as intellectual ability with simple sensory and motor tests bordered on the absurd" (Kaufman, 2009). He proved to be correct when it was shown that Galton's test had almost no correlation to any meaningful criteria of intelligence (Sharp, 1899; Wissler, 1901). Instead, Binet developed a test that evaluated abilities such as memory, judgement, reasoning, and social comprehension. Binet's test was created at the request of the French minister of public instruction in 1904 to differentiate intellectually disabled children from normal children in public schools (Sattler, 2008). By taking into careful consideration linguistic and cultural differences between France and the United States, the American psychologist Lewis Terman of Stanford University adapted the test for use in the US, creating the now-popular Stanford-Binet Intelligence Scales (Terman, 1916). The Stanford-Binet test was the first test to incorporate a measure of Intelligence Quotient (IQ) (Kaufman, 2009).

During the 1930s, David Wechsler created the Wechsler-Bellevue Intelligence Scale (Wechsler, 1939). He disagreed with the single measure of intelligence produced by the Stanford-Binet test because he believed that intelligence tests should act "as windows to the child's or adult's personality" (Kaufman, 2009). He also believed that the Stanford-Binet focused too heavily on verbal skills, so he divided his test into a verbal scale and a performance scale (which focused on non-verbal abilities). The more recent fourth edition of the Wechsler Intelligence Scale for Children (WISC) measures the test taker's ability on verbal comprehension, perceptual reasoning, working memory, and processing speed (Wechsler, 2003).

In 1983, noting that most intelligence tests at the time lacked a formal grounding in theories of intelligence, Alan and Nadeen Kaufman published the Kaufman Assessment Battery for Children (K-ABC) with the intent of creating an intelligence test that is well-grounded in theory (Kaufman and Kaufman, 1983). The second edition of the K-ABC tests children on five different scales: sequential (*Gsm*), simultaneous (*Gv*), planning (*Gf*), learning (*Gv*), and knowledge (*Gc*) (Kaufman, 2004).

The Raven's Progressive Matrices (RPM) test is unique compared to other intelligence tests because it is composed entirely of one type of problem: matrix completion tasks (Raven, 1938).

John C. Raven, the creator of the RPM, was a student of Spearman's and created the RPM to measure Spearman's g factor of intelligence (i.e., the general factor). As a result, the RPM has become generally accepted as the best measure of Spearman's g factor, and it has also gained use as a measure of fluid reasoning (Kaufman, 2009). For this reason, the experiments described in Chapter 6 deal directly with matrix completion tasks.

Many correlations between performance on intelligence tests and on other tasks have been reported in the literature. IQ has been correlated with school performance (Neisser et al., 1996), job performance (Hunter and Schmidt, 1998), and performance on ability and achievement tests (Naglieri and Bornstein, 2003). In fact, in a study by Deary et al. (2007) that tested over 70,000 students on 25 academic subjects, it was found that "general intelligence contributed to success on all 25 subjects." However Neisser et al. (1996) did note in their study of school performance that IQ only accounted for part of the variation and that "successful school learning depends on many personal characteristics other than intelligence, such as persistence, interest in school, and willingness to study." Nonetheless, their results still support the connection between IQ and intelligence. While their concerns are relevant for interpreting children's scores on intelligence tests, they are not as relevant for the robot used in this thesis because it does not have any concept of persistence or interest. Indeed, the correlations between IQ and intelligence indicate that, at the very least, intelligence tests measure something fundamental about how the human brain creates intelligence, and by applying them to robots we can better understand how to make robots more intelligent.

2.2 Use of Exploratory Behaviors in Animals, Humans, and Robots

While intelligence tests have been shown to measure useful criteria of intelligence, the use of exploratory behaviors has been shown to be a fundamentally important part of intelligence in animals and humans. Reports of the use of exploratory behaviors have appeared in the literature since the late 1800s (Darwin, 1874; Kinnaman, 1902; Romanes, 1882; Slonaker, 1912; Small, 1899). Power (2000) postulated that when an animal is performing exploratory behaviors on an object it is essentially asking "What does *this object* do?"

Lorenz (1996) noted that “a young corvide bird, confronted with an object it has never seen, runs through practically all the inventory of its behavior patterns, except social and sexual ones” and that “the young raven experimenting with feeding behavior on a new object does not want to eat; he wants to know whether it is edible *in principle*. The information acquired by exploratory behavior is *objective* in the most literal sense of the word.” He concluded that exploratory behavior is not done for any overt extrinsic motivation (e.g., hunger), but rather out of curiosity, or to gather more information. Indeed, exploratory behaviors of this nature have been reported in the literature among a large variety of animals, including various zoo animals (Glickman and Sroges, 1966), infant baboons (Westergaard, 1992, 1993), wild rats (Inglis and Shepherd, 1994), and male great titmice (a type of bird) (Verbeek et al., 1994). Weisler and McCall (1976) concluded that exploratory behavior is relatively stereotyped and generally follows a fixed sequence and went on to say that animals partake in “*active physical interaction* for the purpose of discerning what will happen as a consequence of the organism’s interactions with the object or situation.” In fact, Inglis and Shepherd (1994) found that animals will even engage in active exploration at possible risk to their own health (in their case they found that rats would repeatedly press a lever that would cause poison to appear where normally food appeared).

Children have also been shown to make use of exploratory behaviors. Piaget (1952) observed that children, when presented with a novel object, explore it as if thinking “What is this thing? I see it, hear it, grasp it, feel it, turn it over, without recognizing it: what more can I do with it?” Eleanor Gibson (1988) postulated that babies spend much of their first year perceiving the affordances of the world around them using exploratory behaviors. She also stated that “perceiving is active, a process of *obtaining* information about the world. We don’t simply see, we look,” and that, as babies gain more control of their fine motor skills, they can perform a wider variety of actions on objects to learn more about the objects’ properties. In a study of object exploration with 2 juvenile chimpanzees, 1 juvenile bonobo, and 1 human child (all 7 to 12 months old), Vauclair and Bard (1983) found that the human child most frequently manipulated and explored the objects as compared to the apes.

Exploratory behaviors can give a child information about an object’s size, shape, hardness,

Table 2.1: Exploratory behaviors identified by Lederman and Klatzky (1987) that are used in deriving particular kinds of knowledge. This table is a close reproduction of the table that appeared in that paper.

Knowledge About Object	*Exploratory Procedure*
Substance-related properties	
Texture	Lateral motion
Hardness	Pressure
Temperature	Static contact
Weight	Unsupported holding
Structure-related properties	
Weight	Unsupported holding
Volume	Enclosure, contour following
Global shape	Enclosure
Exact shape	Contour following
Functional properties	
Part motion	Part motion test
Specific function	Function test

surface texture, weight, volume, consistency, rigidity, and material composition (Power, 2000). Lederman and Klatzky (1987) identified some specific exploratory behaviors that they postulated are used to derive different types of knowledge about objects. The relationship between the behaviors and the knowledge they reveal as described by Lederman and Klatzky (1987) is shown in Table 2.1. Turvey (1996) extended these further by examining how information about object length, weight, size, and orientation can be estimated by the manner in which subjects wield objects.

Interestingly, correlations have been found between the degree of object exploration and performance on other tasks. Verbeek et al. (1994) found that birds that superficially explored their environment had a harder time adapting to changes in the environment than birds that were slower and more thorough. Several findings have also shown correlations between a child's use of exploratory behaviors and other measures of cognitive development (Power, 2000). Yarrow et al. (1983) found that the amount of exploratory behavior at 6 months predicted scores of cognitive development at 12 months. Messer et al. (1986) found that practice of “sensorimotor skills” at 6 months was predictive of scores of cognitive development at 30 months. In another study, Messer et al. (1987) found that at 30 months active object engagement and the extent of investigation were both positively correlated with scores of cognitive development, whereas the amount of “no engagement” was negatively correlated. Ruff and Dubiner (1987) found that the amount of time 12 month olds spent examining during novelty trials was positively correlated with scores of cognitive development and that the amount of time spent not exploring was negatively correlated. Caruso (1993) found a positive correlation between the breadth of exploratory behavior in 11-12 month olds and success on problem solving tasks. Pedersen and Wender (1968) found that teacher ratings of “sustained directed activity” at age two and a half years was positively correlated with IQ scores on the Wechsler Intelligence Scale for Children at age six and a half years, although this is contradicted by Henderson and Wilson (1991) who found no correlation between exploration of novel objects by preschoolers and scores on the Kaufman Assessment Battery for Children. Nonetheless, all these results taken together yield valuable insights. While it is important not to confuse correlation with causation, these studies do show that exploratory behaviors are fundamentally linked with children’s ability to understand the properties and affordances of objects.

Recently, there has been a lot of work in robotics that makes use of exploratory behaviors. Fitzpatrick et al. (2003) used a robot to explore objects by pushing them with its end-efector. Krotkov et al. (1997) and Torres-Jara et al. (2005) both used exploratory tapping motions to detect the acoustic properties of objects. Natale et al. (2004) used proprioceptive data from the robot’s hand grasping objects to perform object recognition. Stoytchev (2005) used exploratory behaviors to learn tool affordances. Other work from the Developmental Robotics Lab at Iowa State University has used exploratory behaviors to perform object recognition (Sinapov et al., 2011a), solve the odd-one-out task (Sinapov and Stoytchev, 2010b), and perform object category recognition (Sinapov and Stoytchev, 2011). Sinapov and Stoytchev (2010a) even showed a theoretical link between the use of exploratory behaviors and the concept of boosting in machine learning.

The exploratory behaviors used in this thesis were developed over the course of multiple previous experiments conducted in the lab. The first iteration of the set of exploratory behaviors, described in Sinapov et al. (2008), was used to perform object recognition based solely on auditory features. In that paper the robot performed the behaviors *grasp*, *drop*, and *push* on each object. In a follow-up paper, Sinapov et al. (2009) added the *shake* and *tap* behaviors. These behaviors were modified in Bergquist et al. (2009) to *lift*, *shake*, *drop*, *crush*, and *push* in order to perform object recognition using only proprioceptive features. In Sinapov et al. (2013), additional behaviors were added, resulting in *look*, *grasp*, *lift*, *hold*, *shake*, *drop*, *tap*, *poke*, *push*, and *press*. These were used to perform object category recognition over a set of 100 objects. Over time the set of exploratory behaviors has expanded from initially only 3 behaviors, to 5 behaviors, and finally to 10 behaviors. In this thesis we add another behavior, *rattle*.

2.3 Measuring Object Similarity

The ability to measure the similarity between a pair of objects is fundamentally important to solving multi-object tasks. An experiment by McPherson and Holcomb (1999) investigated this by examining event-related brain potentials. Participants were shown a picture of an object, then a picture of another object from one of three categories: related [to the first object], moderately related, or unrelated. The electroencephalogram (EEG) results showed that across all participants, there was a large negative spike in the N400 family of potentials in the participants' brains shortly after being shown the second picture. The study found that the magnitude of the spike was related to the similarity between the two objects in the pictures. This suggests that, at least at some level, the brain computes a quantitative measure of how similar two objects are.

In another study with 1- to 3-year-olds, Sugarman (1981) found that the order in which children interact with objects tends to be influenced by the class and perceptual similarity of the current object to the previously explored object. Additionally, it was observed that the older children relied less on the class of the object to pick the next object and more on perceptual similarity. They concluded that "classification based on conceptual comparisons of the items being selected may emerge over the second and third years." This shows that not only are very young children able to compare the perceptual similarity between objects, but also that they actively use it as a basis for classification and interaction with objects. Because this behavior appears spontaneously at such a young age, it suggests that comparing objects based on perceptual similarity can be very useful for interacting in human environments.

Several studies have demonstrated that robots can measure perceptual as well as functional object similarities for a variety of tasks (Nolfi and Marocco, 2002; Natale et al., 2004; Nakamura et al., 2007; Takamuku et al., 2008; Sun et al., 2010; Sinapov and Stoytchev, 2010b). The ability to measure the similarity between two objects is extremely useful for tasks such as category recognition and object grouping. Several studies (Nakamura et al., 2007; Sinapov et al., 2011a) have used unsupervised approaches for object categorization, in which objects were categorized based on the similarity of their perceptual features. Their results showed that when the robot was allowed to use all of its sensory modalities, its object categorizations closely resembled the human-provided ones. This suggests that allowing robots to perceive more features about objects can improve their ability to detect similarities between objects.

2.4 Solving Multi-Object Tasks

Perceptual reasoning tasks on intelligence tests are most commonly posed as multi-object tasks. For example, the perceptual reasoning section of the WISC-IV consists of four subtests: *block design*, where children must put blocks in a specified pattern; *picture concepts*, where children are given pictures and asked to determine which pictures go together; *matrix reasoning*, where children are given an array of pictures with one missing and asked to fill in the missing one; and *picture completion*, where children are shown pictures of common objects with a part missing and must identify the missing part (Wechsler, 2003). All of these tasks require the test taker to be able to understand and relate multiple objects at once in order to solve the task (for *picture completion* the child must understand the relationship between the different parts of the object in order to understand what is missing). The RPM test, one of the best measures of perceptual reasoning, consists exclusively of multi-object tasks (Raven, 1938). This indicates that the best way to test perceptual reasoning ability is by posing tasks that involve understanding and relating multiple objects at once.

The ability of infants to relate objects has been well documented in the literature. For example, infant baboons have been shown to exhibit relational behaviors on objects when exploring and playing with them (e.g., placing objects in cups) (Westergaard, 1992, 1993). Infant chimpanzees have also been shown to exhibit object–object behavior at a fairly young age (e.g., inserting objects, stacking objects) (Hayashi and Matsuzawa, 2003). Other work by Hayashi et al. (2006) has shown that both infant gorillas and bonobos engage in relational activity between multiple objects and that the infant gorillas had a tendency to bang objects together. Younger (1985) showed that ten-month-old human infants can form object categories and determine the variants and invariants of the objects within a category, and based on that information they can determine the inclusion of a novel object in the given category. These studies show that relational activity is quite common when human and ape infants manipulate objects.

There have been multiple studies that have demonstrated a robot's ability to perform multi-object tasks. Several studies (Nakamura et al., 2007; Sinapov et al., 2011a, 2013) have shown how a robot can solve the object categorization task. Other studies (Griffith et al., 2008, 2009, 2010; Griffith and Stoytchev, 2010; Griffith et al., 2011, 2012a,b) showed how a robot can learn container and non-container categories for novel objects. In those experiments, the robot would frequently place a smaller object in a specific spot relative to the object that it was testing (e.g., it might attempt to place the small object inside the other object), and then it would observe the co-movement of the two objects while it interacted with them in order to determine if the larger object was a container. The next four sub-sections go into more detail on specific tasks.

2.4.1 The Odd-One-Out Task

One standard type of problem on intelligence tests is the odd-one-out task. Test takers are presented with a group of objects and must select the object that does not belong with the rest. An example odd-one-out task is shown in Figure 2.1. Sinapov and Stoytchev (2010b) showed how a robot can solve odd-one-out tasks. Much of that work has motivated the work done in this thesis, and so their methodology will be briefly explained here. The robot first interacted with a set of 50 household objects by performing the *lift*, *shake*, *drop*, *crush*, and *push* behaviors

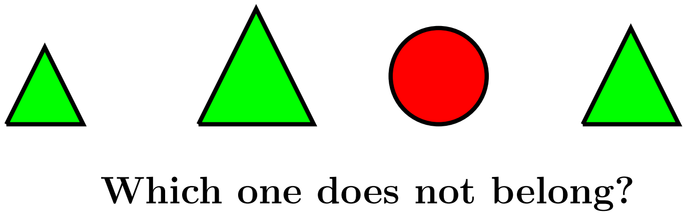


Figure 2.1: An example odd-one-out task. The correct answer is the red circle because both its color and its shape are different from those of the other three objects.

on each object while recording from both its auditory and proprioceptive sensory modalities. It then trained two Self-Organizing Maps (SOMs), one for audio and one for proprioception, and used them to discretize the raw sensory data into strings. The robot then computed a similarity matrix for each sensorimotor context (where a context is defined as the combination of a sensory modality and an exploratory behavior, e.g., *audio-drop*) by computing the global-alignment distance (Navarro, 2001) between the strings for each pair of objects. The similarity matrix from one of the tasks posed to the robot is shown in Figure 2.2a. Figure 2.2b shows the Isomap embedding (Tenenbaum et al., 2000) of the similarity matrix. The figure shows that the red hat is the least similar to the rest of the objects, and thus it is the "odd-one-out."

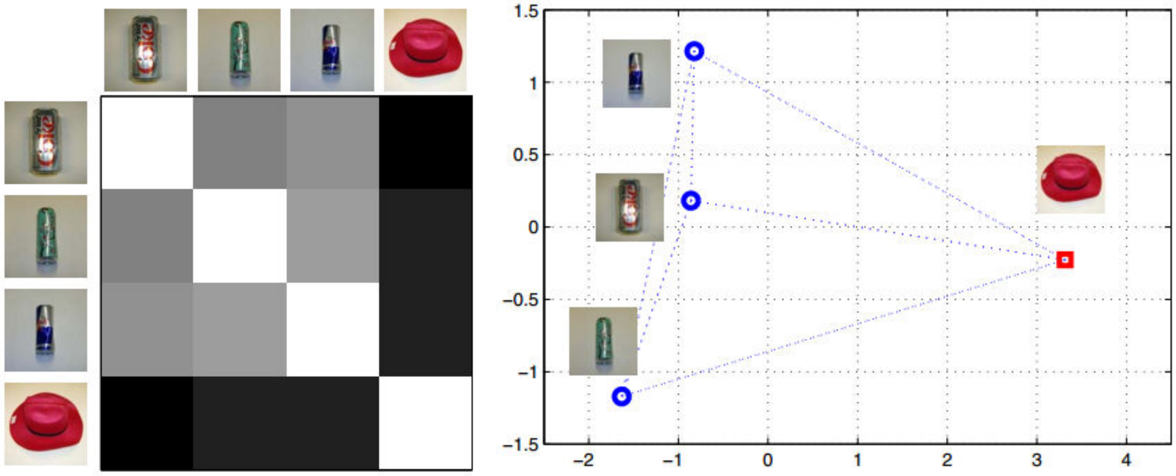
Given a set of similarity matrices, one for each context, the robot first had to combine the matrices into a consensus matrix $\mathbf{W}$. The robot computed each entry in the matrix as follows:

$$\mathbf{W}_{ij} = \sum_{c \in \mathcal{C}} \alpha_c \times W_{ij}^c$$

where $\mathcal{C}$ is the set of all sensorimotor contexts, α_c is the weight assigned to context c, and W_{ij}^c is the similarity between objects i and j in context c. The weights assigned to each context were determined using two different schemes: uniform weighting and weighting based on the object recognition accuracy of the robot when using information from only that context. Next, the robot had to use this consensus matrix to solve odd-one-out tasks posed by the experimenters. Given a set of objects $\mathcal{T}$, the robot selected the object $i \in \mathcal{T}$ that maximized the following objective function:

$$q(\mathcal{T}, i) = \alpha_1 \sum_{j \in \mathcal{T}/i} \sum_{k \in \mathcal{T}/i} \mathbf{W}_{jk} - \alpha_2 \sum_{j \in \mathcal{T}/i} \mathbf{W}_{ij}$$

where α_1 and α_2 are normalization constants. Sinapov and Stoytchev (2010b) described this function as follows: "The first term captures the pairwise object similarity between the re-



(a) An example similarity matrix.

(b) The Isomap embedding of an example odd-one-out task.

Figure 2.2: An example odd-one-out task posed to the robot by Sinapov and Stoytchev (2010b): a) the similarity matrix (lighter colors indicate higher similarity); b) the Isomap embedding of the matrix shown in a). It is clear from both of these that, in this task, the red hat is the odd object. Reproduced from that paper.

maining objects in $\mathcal{T}$ (i.e., after i is removed from $\mathcal{T}$). The second term captures the similarity between the selected object i and the remaining $K-1$ objects in $\mathcal{T}$."

The robot in (Sinapov and Stoytchev, 2010b) was able to find the odd object with varying degrees of success, depending on the category of the three most similar objects. For *pop cans* it was able to pick the object that was not a pop can with 100% accuracy, but for *objects with contents* it was only able to pick the object without contents 66.71% of the time using the best context. Nonetheless, Sinapov and Stoytchev (2010b) concluded that "the robot's choice for the odd object was consistent with human-defined object categories" and that "sensorimotor interaction can capture many of the physical properties of objects that define an object category." This shows that the framework developed in that paper can be very useful for perceiving physical properties of objects and solving tasks based on those properties. This framework will serve as the basis for the experiments described in this thesis. The experiments described in Chapter 4, which solve the object pairing task, are a direct extension of the work Sinapov and Stoytchev (2010b) did on solving the odd-one-out task. The experiments described in Chapters 5 and 6, which solve the object ordering and matrix completion tasks

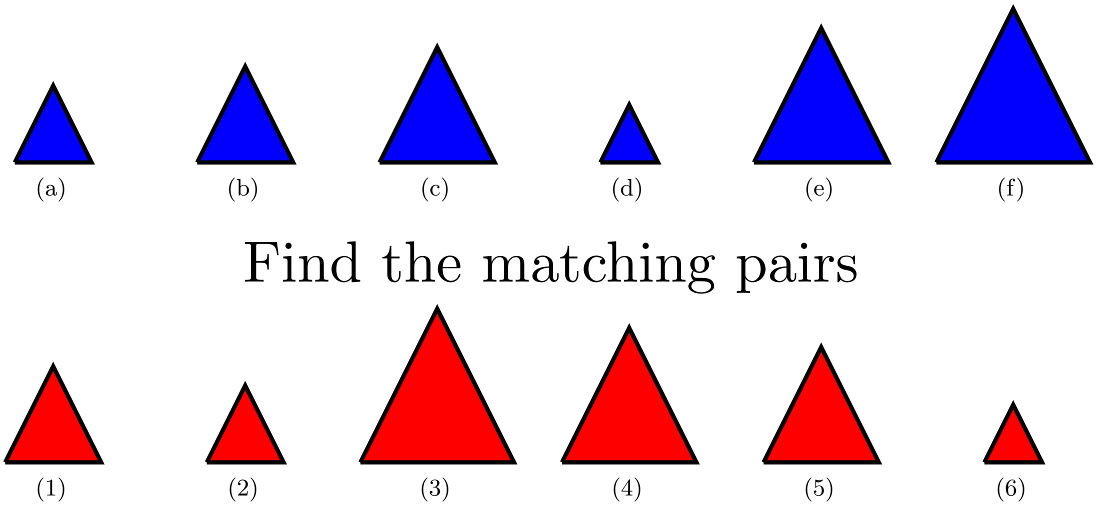


Figure 2.3: An example matching task. The objects in each of the two sets vary by size. For each object in the first set (blue objects) there is exactly one object in the second set (red objects) that has the same size. The goal is to find the matching pairs of objects. The correct matching is as follows: (a) goes with (2), (b) goes with (1), (c) goes with (5), (d) goes with (6), (e) goes with (4), and (f) goes with (3).

respectively, make slight alterations to the framework. The next three subsections describe the tasks addressed in this thesis and previous work related to them.

2.4.2 The Object Pairing Task and The Montessori Method

The object pairing task is another common task on intelligence tests. In this task the test takers are presented with two groups of objects and asked to find matching pairs of objects between the two groups. It is also common for there to be just one group of objects and test takers must find matches within the group. *Picture concepts* is a matching task that appears on the fourth edition of the Wechsler Intelligence Scales for Children (WISC). In this task, children are shown two or three rows of images and must match the images between the rows based on which ones go together (Wechsler, 2003). Indeed, the ability to pair objects is one of many important tasks used to measure intelligence on these tests.

A variant of the object pairing task was used by Daehler et al. (1979), who used both real objects and pictures of objects in their experiments. They found that children around the age of two are able to correctly match both pictures and objects to sets of pictures or objects. One interesting result of their experiment was that the children performed significantly better on tasks where they were asked to match an object to a set of objects, versus picture to object, object to picture, or picture to picture matching. They suggested that this was due to the ability of the children to perceive the objects from multiple angles, thus giving them more reliable information about the objects than they could extract from the pictures. Other studies have shown that infants can identify object pairs and group objects into categories. A study by Leslie and Chen (2007) demonstrated that eleven-month-old infants can individuate pairs of objects only when there is a large amount of physical similarity between the objects in the same pair (in this study they used identical objects) and a large physical difference between objects of different pairs. These studies together indicate that object pairing can be a useful way to test the ability of a test taker to perceive differences between objects and use this information to accomplish some goal.

The object pairing task is also used as part of the Montessori methodology of teaching (Pitamic, 2004). The Montessori method is an alternative style of education for children. It was developed in the early 1900s by the Italian educator Maria Montessori (1870-1952) (Montessori, 1912). The Montessori method is characterized by a special set of educational materials and student-directed learning activities (Montessori, 1912; Lillard, 2008; Lillard and Else-Quest, 2006). One of its core principles is that of *embodied cognition*, tying movement of the body and learning together. It focuses on stimulating the development of different skill sets, including sensory development, language development, and numeracy skills. Most Montessori tasks require that the children actively touch, move, relate, and compare objects (Lillard, 2008). An example Montessori task is shown in Figure 2.4.

Recent studies have found that students educated using the Montessori method often outperform students educated by traditional methods. For example, one study found that middle school students from Montessori schools had higher intrinsic motivation when it came to academic activities as compared to students from traditional schools (Rathunde and Csikszentmihalyi, 2005). This suggests that the Montessori method is more effective at fostering learning in young children than the traditional methods. This conclusion was supported by another study (Lillard and Else-Quest, 2006), which found that, by the end of kindergarten, Montessori

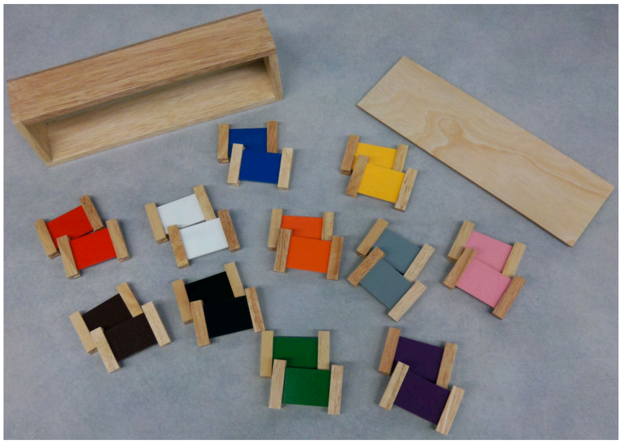


Figure 2.4: A task from the Montessori method. In this task, there are 22 tiles, made up of 11 pairs of the same color. The task is to match the tiles based on their color.

students outperformed traditional students on standardized tests of reading and math and also showed more advanced social skills and executive control. This suggests that the objects used by the Montessori method may provide a good environment for testing not only object pairing skills, but also for determining how a robot might learn to solve these kinds of tasks. We will test this hypothesis in Chapter 4.

2.4.3 The Object Ordering Task

Object ordering tasks appear on several intelligence tests. In these tasks the test takers are given a set of objects and asked to place them in order. In the Intelligence and Development Scales (IDS) test (Hagmann-von Arx et al., 2008), children are asked to sort lines of varying length. In a more common test, the Wechsler Intelligence Scale for Children (WISC) (Kaufman, 1994), participants are asked to place images from a story into a logical sequence. While it is not currently feasible for a robot to understand the events taking place in an image, these two tests show that, given an understanding of the objects, knowledge of how to order them is a strong indicator of intelligence.

An alternative formulation of this task is the order completion task. In this version, the test taker is given an already ordered set of objects and asked to pick another object that completes the ordering. It is this formulation of the object ordering task that we will use in Chapter 5.

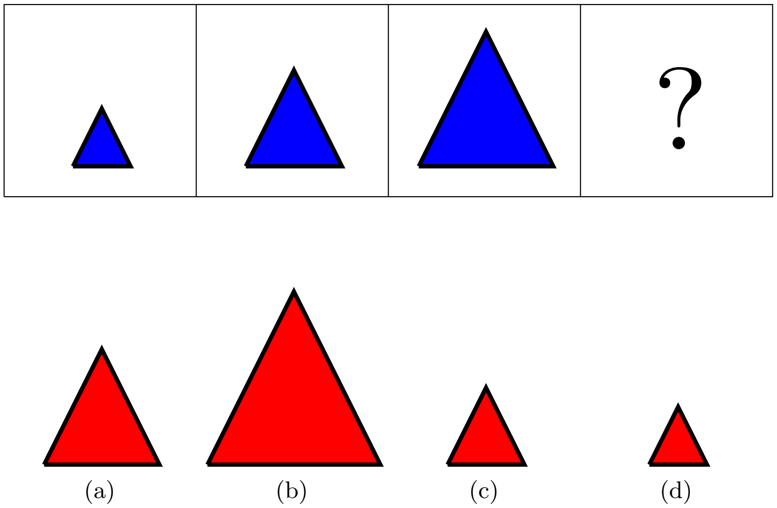


Figure 2.5: An example order completion task. The objects in the top row (the blue triangles) are ordered by size, increasing from left to right. The goal is to pick one of the four candidate objects shown in the second row such that it completes this order. The correct answer is (b) because it increases in size in a manner consistent with the first three objects in the ordering.

An example order completion task is shown in Figure 2.5.

Several studies have shown that young children have a fundamental understanding of the concepts underlying ordering. Graham et al. (1964) found that children between the ages of 2 and $4\frac{1}{2}$ can easily judge an object as “big” or “small” when compared to another object. Two studies by Ebeling and Gelman (1988, 1994) found similar results. Interestingly, all three studies found that children were much better able to judge an object as “big” or “small” when they compared it with immediately viewable objects as opposed to making the judgment based on the object’s absolute size (Graham et al., 1964), its normative size (i.e., how big it is compared to the typical object in the category) (Ebeling and Gelman, 1988), or its functional size (i.e., how big it is in relation to the function it is to perform) (Ebeling and Gelman, 1994). Thus, the ability to compare an object to other directly viewable objects is a prerequisite for successfully performing the task of ordering. Furthermore, this type of comparison is used more often than other types of comparisons (e.g., absolute, normative, or functional) at such an early age, which indicates that it is an important aspect of intelligence.

A study by Sugarman (1981) found that the order in which children interact with objects is highly influenced by the objects' perceptual similarity. She found that 1- to 3-year-old children relied both on the class of the objects and on the perceptual similarity, but that older children tended to disregard the class more often than younger children and relied more heavily on just the perceptual similarity of the objects. In Chapter 5 the robot uses the perceptual similarity of the objects to each other in order to discover the underlying ordering in a group of objects and then to complete the ordering.

To the best of our knowledge, the object ordering task has not been previously done in robotics, although in machine learning, the problem of ranking (i.e., placing a set of data in the correct order) has been well studied (Chapelle et al., 2011). There are many algorithms that can solve ranking with a high degree of accuracy. Ranking, however, is not the same as object ordering. Standard ranking methods are often supervised or at least semi-supervised, but object ordering tasks on intelligence tests offer very little in the way of training. Instead, they rely on the test taker's ability to perceive the salient features of the task and use those to order the objects. Additionally, standard ranking algorithms often perform poorly when the number of objects is small, but object ordering tasks on intelligence tests frequently utilize only a small number of objects at a time. Thus, ranking algorithms are not well-suited for solving object ordering tasks.

2.4.4 The Matrix Completion Task

Matrix reasoning skills are well studied in psychology. In the matrix completion task, the test taker is shown a grid of objects with one object missing and must infer the patterns in the grid in order to deduce the missing object. The most well-known matrix completion test, the Raven's Progressive Matrices (RPM) test (Raven, 1938), is known to predict performance on a wide range of reasoning tasks (Prabhakaran et al., 1997). The RPM test is composed exclusively of matrix completion tasks, and more recent versions of popular intelligence tests have begun to incorporate matrix reasoning into their tests. For example, the third version of the Wechsler Abbreviated Scale of Intelligence (WASI) added a section on matrix reasoning (Wechsler, 1997). A study by Fuchs et al. (2008) found that performance on the WASI's matrix reasoning section was predictive of third-graders' problem solving abilities. Another study by

Dugbartey et al. (1999) found a correlation between performance on the WASI matrix reasoning section and the Halstead Category Test, which measures "complex spatial abstract reasoning and the use of conceptual rules in reasoning with ratios and proportions." Both of these suggest that performance on matrix completion tasks is indicative of performance on other complex reasoning tasks.

Dugbartey et al. (1999) found that, on the WASI matrix reasoning section, "elementary visuoperceptual abilities may be a necessary, but not sufficient, requirement for success on reasoning with matrices." This suggests that an ability to understand the objects in the matrix is fundamental to being able to solve the task. This was reinforced by a study by Richardson (1991), which found that children performed significantly better on the RPM test when problems were altered to use more meaningful symbols while the underlying structure of the problem remained the same. Richardson found that after converting from using abstract symbols (e.g., lines and circles) that were difficult for the children to understand, to contextually meaningful symbols (e.g., cars and people) that were easy for the children to understand, the children were better able to perceive the patterns present in the matrices. Previous work from our lab has dealt extensively with developing our robot's ability to understand the physical properties of objects (Sinapov et al., 2011a; Sinapov and Stoytchev, 2011, 2010a; Sinapov et al., 2013). In Chapter 6, similar to Richardson (1991), we will utilize physical objects to pose matrix completion tasks to the robot while maintaining the general structure that underlies matrix reasoning problems.

Carpenter et al. (1990) proposed a taxonomy of rules that, according to them, govern the patterns in the rows and columns of the matrices in the RPM test. The rules they proposed were (as restated by Little et al. (2012)):

- *constant*: the same value occurs throughout a row, but changes down a column
- *increment*: a quantitative increment between adjacent entries in an attribute such as size, position, or number
- *decrement*: a quantitative decrement between adjacent entries in an attribute

Table 2.2: Rules identified by previous work and used in the experiments in Chapter 6.

Carpenter et al. (1990)	Little et al. (2012)	This Thesis
Constant in a row	Constant	**Constant**
Quantitative pairwise progression	Increment Decrement	**Increment** **Decrement**
Figure addition or subtraction	Logical AND Logical OR	N/A
Distribution of 3	Permutation	**Permutation**
Distribution of 2	Logical XOR Distribution of 2	N/A

- *permutation*: a permutation of different values across a row for a categorical attribute such as figure type
- *logical AND*: the third object in a row is the logical AND of the first two
- *logical OR*: the third object in a row is the logical OR of the first two
- *logical XOR*: the third object in a row is the logical XOR of the first two
- *distribution of 2*: two objects in a row have an identical value and the third object has a different (usually null) value

They claimed that all but 2 of the 60 problems on the Advanced Progressive Matrices test (a variant of the RPM) could be stated as a combination of some subset of these rules. In Chapter 6, we will use the rules *constant*, *increment*, *decrement*, and *permutation*, based on the rules described in Carpenter et al. (1990), to generate matrix completion tasks for the robot to solve. Table 2.2 shows the mapping of the rules from the taxonomy that Carpenter et al. (1990) derived, to the rules stated by Little et al. (2012), to the rules used in this thesis. We chose not to use any of the rules based on logical operators and the *distribution of 2* rule due to the constraints imposed by utilizing physical objects rather than images as the entries in each matrix completion task.

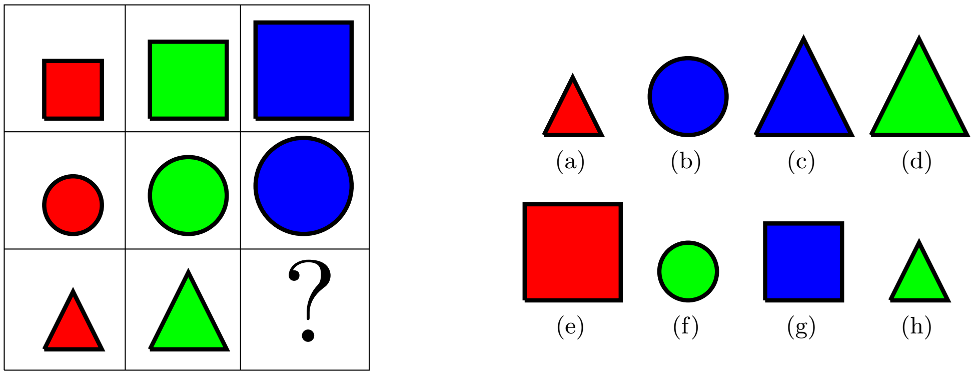


Figure 2.6: An example matrix reasoning task. The correct answer is (c) because it is the same shape as the other objects in the bottom row, it is increasing in size over the previous object in its row, and it completes the permutation over the colors in the bottom row. Looking down the columns, it is the same size and color as the other objects in its column and it completes the permutation over the shapes in the last column.

An example task that uses these rules is shown in Figure 2.6. The matrix on the left side of the figure exhibits multiple instances of the rules: *constant* in shape across the rows; *increment* in size across the rows; *permutation* for color across the rows; *permutation* for shape across the columns; *constant* in size across the columns; and *constant* in color across the columns. Given this, the only object that can be placed at the question mark and maintain consistency with these rules in the matrix is object (c).

2.5 Intelligence Tests in AI

The field of Artificial Intelligence has a long history of building specialized systems for solving tasks that are commonly associated with human intelligence. A large number of these systems has focused on solving games and puzzles that are challenging for many humans. Probably the most famous example of such a system is IBM's *Deep Blue* computer, which defeated then-reigning World Chess Champion Gary Kasparov in 1997 (Campbell et al., 2002). Deep Blue was the first computer to ever beat the best human chess player. More recently IBM developed the *Watson* computer to compete on the popular television game show *Jeopardy* (Ferrucci et al., 2010). It defeated then-best contestant Ken Jennings, who famously said, once defeat was inevitable, “I, for one, welcome our new computer overlords” (Markoff, 2011). *Watson* not only represented the first AI system to compete in an open-ended question game show, but also the first one to beat the best human players. There have also been AI programs that have performed exceptionally well at checkers (Schaeffer et al., 1996), crossword puzzles (Littman et al., 2002), and sudoku (Sato and Inoue, 2010). But all of these AI systems have one fundamental problem: they are all designed to work for specific tasks and do not generalize well to other tasks.

Turing Test, the most well-known test in artificial intelligence, was designed to facilitate the development of AI systems that can generalize their skills to many tasks (Turing, 1950). In this test, a computer and a human have a teletyped conversation (that is, a conversation purely through text) with a human judge. A computer is said to have passed the test if it can make itself indistinguishable from the human to the judge. Ideally this means that the computer would have to be as intelligent as the human in order to make this possible. Although well-known, this test has been widely criticized as an inadequate measure of intelligence. Hernández-Orallo and Dowe (2010) noted that the Turing Test presents "several problems as a machine intelligence test: the Turing Test is anthropomorphic (it measures humanity, not intelligence); it is not gradual (it does not give a score); it is not practical (it is increasingly easy to cheat and requires a long time to get reliable assessments); and it requires a human judge." Legg and Hutter (2007) stated that "even small things like pretending to be *unable* to perform complex arithmetic quickly and faking human typing errors become important, something which clearly goes against the purpose of the test." It is apparent from these criticisms that the Turing Test is not a good measure of intelligence, at least for machines.

Amant and Wood (2005) proposed an alternative called the "Tooling" Test based on the Turing Test. In this test, a robot would perform various tasks with tools, and a human observer would have to decide if the robot was being controlled by a computer program or by an experienced human teleoperator. The tasks they proposed were based on experiments showing tool-use in humans and animals. The tasks, from least complex to most complex, are: *simple tool use*, *tool construction*, *non-transparent tool use*, and *collaborative tool use*. While this test presents a novel and interesting method for testing intelligence in robots, some of the criticisms of the Turing Test still apply to this test. For example, if any mannerisms developed in the way in which human operators controlled the robot through the teleoperation interface, then these could be used to easily distinguish between a human operator and a program without regard to the program's intelligence (similar to the way spelling errors can be used to distinguish between a human and a computer in the regular Turing Test). Nonetheless, the "Tooling" Test still poses an interesting problem for robotics researchers to solve.

Research at the intersection of artificial intelligence and theoretical computing has also attempted to remedy some of the problems of the Turing Test by developing tests based on algorithmic complexity theory. Legg and Hutter (2007) developed a mathematical model for what they call "Universal Intelligence." Essentially, an agent's universal intelligence is measured as the amount of reward it is expected to attain in a variety of environments, inversely weighted by the complexity of each environment. Hernández-Orallo and Dowe (2010) proposed the "Anytime Universal Intelligence Test," based on ideas similar to the ones proposed by Legg and Hutter (2007). In their work, the test is composed of environments that are dynamically selected based on the performance of the agent. The idea is that, if an agent is performing well in environments of a certain level of complexity, then it should be tested in more complex environments, and vice versa. In other words, the test adapts to the skills of the agent.

A fundamental problem with this method of testing intelligence is that it is not well-grounded in the psychology literature. In other words, when a new intelligence test is developed, it must be shown to correlate with other measures of intelligence in order to be considered valid (Kaufman, 2009). Insa-Cabrera et al. (2011) tested the intelligence test proposed by Hernández-Orallo and Dowe (2010) on both humans and the reinforcement learning algorithm Q-learning. They defined a set of grid environments where the test taking agent had to accrue positive reward by following an agent named *good* and avoid negative reward by avoiding an agent named *evil*. They compared the performance of the reinforcement learning algorithm to that of the human subjects, but they did not compare human subject performance on this test with human subject performance on other, validated intelligence tests. In fact, to our knowledge, the test proposed by Hernández-Orallo and Dowe (2010) has not been correlated with any validated intelligence test or other meaningful criteria of intelligence.

There have been other researchers who have developed computational solutions to tasks that occur on human intelligence tests. Sanghi and Dowe (2003) proposed formulations for tasks from common intelligence tests that a program could solve (e.g., odd-man-out, sequence completion). They even developed a program to solve some of these tasks, but the program relied heavily on a look-up table in order to detect and complete patterns, decreasing its usefulness as a means of understanding intelligence. There have also been multiple attempts at computational solutions for the Raven's Progressive Matrices (RPM) test published in the literature (Carpenter et al., 1990; Little et al., 2012; Lovett et al., 2010; Rasmussen and Eliasmith, 2011; Kunda et al., 2010; Cirillo, 2010). The RPM intelligence test is a natural choice for researchers in AI because it is one of the few tests that requires little background knowledge (e.g., it does not require understanding of language). Most of the computational solutions, though, utilize hand-coded features, rather than real data from sensors. The algorithm developed by Kunda et al. (2010) was the only one that did not utilize hand-coded features. Instead, they relied on complex mathematical transformations between images of the objects in the RPM to solve the tasks. None of these computational solutions, though, used a robot. To the best of our knowledge, the experiments described in Chapter 6 of this thesis represent the first attempt at solving matrix completion tasks using an embodied approach.

2.6 Summary

Intelligence tests have evolved significantly since the first modern test was proposed by Sir Francis Galton. More recent tests cover mentally rigorous areas such as verbal comprehension, working memory, and perceptual reasoning. These tests have been shown to have significant correlations with meaningful criteria of intelligence (e.g., as grade point average or job performance), indicating that they measure intelligence at least to some extent. Additionally, the use of exploratory behaviors in animals and humans has also been shown to correlate with meaningful criteria of intelligence. This indicates that exploratory behaviors can be very useful for performing tasks that require some degree of intelligence. For this reason, the robot employed in this thesis used exploratory behaviors to learn about objects and to solve perceptual reasoning tasks.

Perceptual reasoning tasks are most commonly posed as multi-object tasks. The test taker is given a set of objects, arranged in a specific configuration, and asked to pick from a set of candidate solutions. Tasks such as the odd-one-out task and the object pairing task require the test taker to provide additional structure to the given objects. Other tasks such as the order completion task and the matrix completion task require the test taker to fill in the missing object. There is much work in the psychology literature showing that infants and young children are proficient in many of the skills required to solve these tasks. These skills appearing at such an early age suggests that they are an important part of human environments and that these tasks are a good way to measure their development. In this thesis we test this hypothesis by evaluating our robot's ability to solve the object pairing task, the order completion task, and the matrix completion task.

CHAPTER 3. EXPERIMENTAL PLATFORM

3.1 The Robot

The robot used in this thesis is an upper-torso humanoid robot. It is shown in Figure 3.1. The robot has two arms which are mounted in an anthropomorphic configuration. The robot was designed to be able to do many of the same tasks that infants and young children can do, and in this setup the robot has similar capabilities to an infant sitting in a high-chair. It can explore and manipulate the objects placed on the table in front of it. In other words, the table shown in Figure 3.1 acts as the interaction surface for the robot and was used in all experiments described in this thesis.

The two arms of the robot are 7-DOF Barrett Whole Arm Manipulators (WAMs). Each arm has an attached Barrett hand. The arms are mounted opposite each other on top of a large metal cylindrical stand. The robot's head has 9 degrees of freedom and is mounted above the arms on top of the stand. The stand is attached to a metal cart. The cables attached to the robot's head and arms run through the hollow interior of the metal stand and connect to the electronics cabinet on the back of the cart. In the cabinet are the computers that control the robot's arms, audio equipment for processing auditory data from the robot's microphones, and power boxes for the robot's arms and head. Plastic covers are attached to the robot's head and chest in order to improve its appearance.

Figure 3.1 shows the robot with the table in front of it and the electronics cabinet behind it, covered by a black cloth. This setup was used for the experiments described in Chapters 4 and 5, which primarily used the robot's left arm. During the experiments described in Chapter 6, the robot's left arm required maintenance and as a result was removed along with the plastic chest covers. Since the experiments only required one arm, the robot's right arm was used instead.

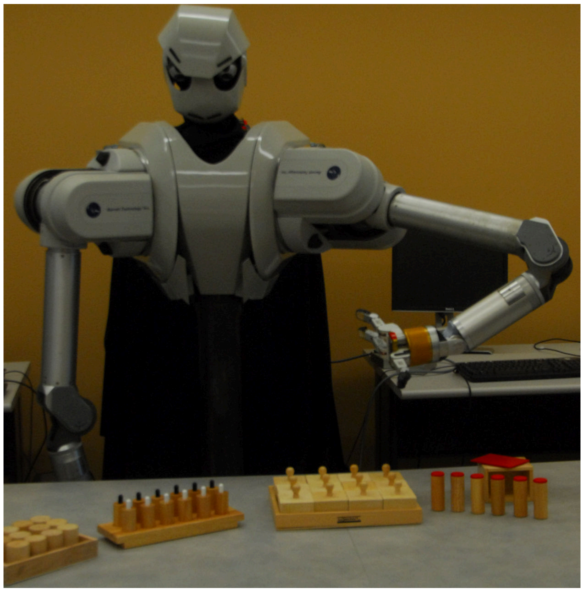
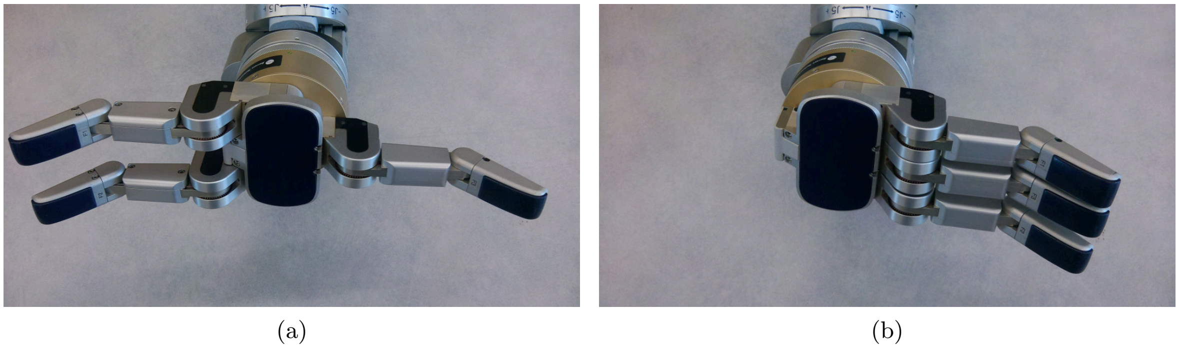


Figure 3.1: The upper-torso humanoid robot. The robot is shown here with some of the Montessori objects that were used in the experiments described in Chapters 4 and 5.

This did not affect the robot's performance.

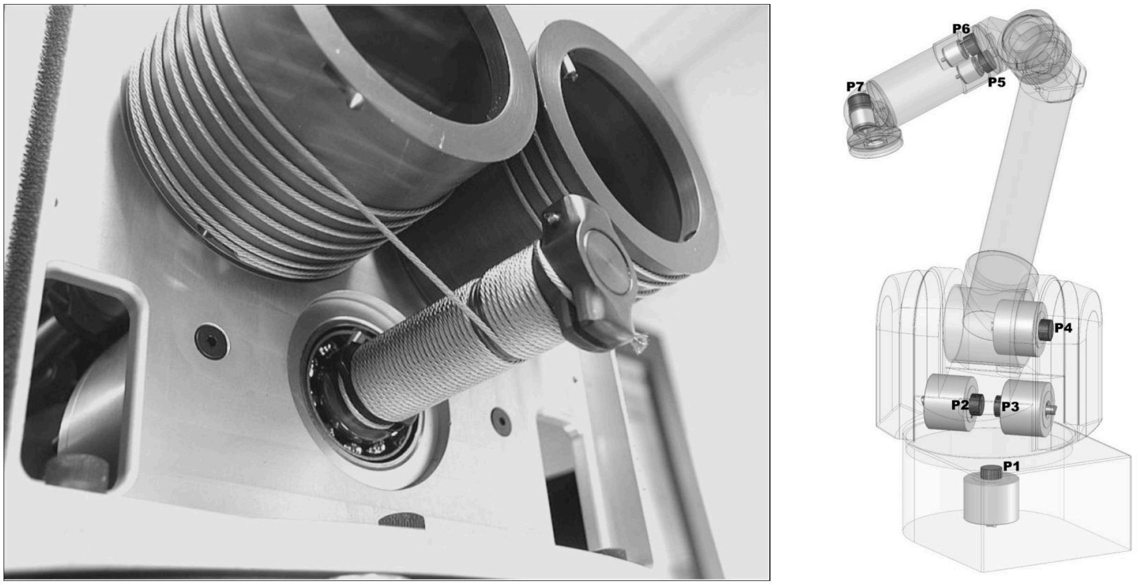
Figure 3.2 shows the robot's right hand. It has three fingers, all of which have two joints and can open and close fully, though the two joints in each finger are controlled by a single motor. The fingers can also change spread, that is, the two outermost fingers can pivot around the palm of the hand and either be on the same side as the other finger or on the opposite side. The hand with the finger spread open is shown in Figure 3.2a and the hand with the finger spread closed is shown in Figure 3.2b. The right hand of the robot is equipped with touch



(a)

(b)

Figure 3.2: The Barrett hand attached to the robot's right arm. The image on the left shows the hand with the spread open; the image on the right shows the hand with the spread closed.



(a) (b)

Figure 3.3: The drive mechanism of the Barrett WAM. The left image shows the cable-and-cylinder drive for one of the joints; the right image shows the layout of the pucks in the WAM. Adapted from Rooks (2006).

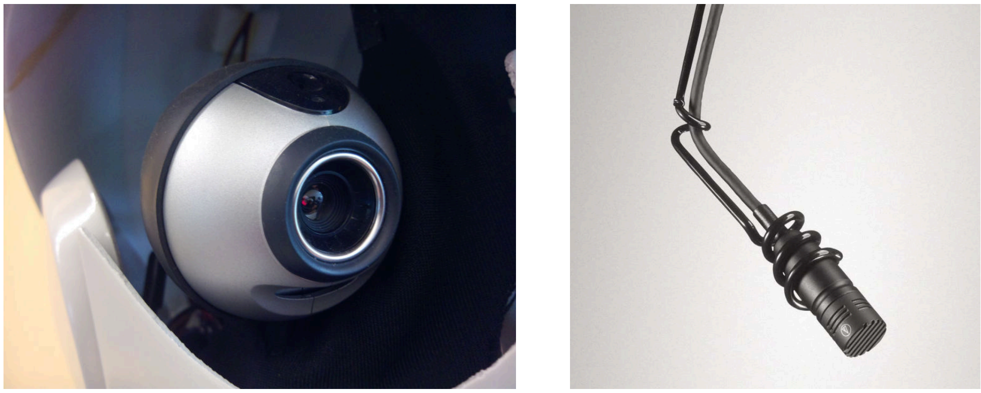
sensors, which are visible in the images as the blue pads on the robot's fingertips and palm. The robot's left hand is an older model that does not have touch sensors.

Figure 3.3 shows the internal design of the WAM. The WAM is cable-driven by servo-controllers called pucks. Part of the cable-and-cylinder drive of the WAM is shown in Figure 3.3a. The layout of the seven motor controllers (pucks) that control the seven joints of the WAM is shown in Figure 3.3b. Pucks 1, 2, and 3 control the orientation of the robot's shoulder joints, puck 4 controls the robot's elbow, and pucks 5, 6, and 7 control the orientation of the robot's wrist joints.

3.2 The Robot's Sensors

The robot has multiple sensors that allow it to perceive data from three different sensory modalities: proprioception, audio, and vision.

For proprioception, the robot used the joint-torque sensors built into the motor controllers (pucks). At any point in time, this allowed the robot to measure both the position of each joint



(a) RGB Logitech Webcam

(b) Audio-Technica U853AW microphone

Figure 3.4: The robot's sensors: a) one of the robot's 2 RGB Logitech Webcams; b) one of the robot's 2 Audio-Technica U853AW microphones.

and the torque being applied to each joint. In the experiments described in this thesis, the robot sampled these values at 500 Hz. The joint positions were used to keep the robot's arm on a pre-specified trajectory. The joint torques were used as the proprioceptive data for the algorithms described in this thesis. They were recorded as a 2-dimensional array, where one dimension varied over time and the other varied over the different joints in the robot's arm. An example of a joint torque recording is shown in Figure 3.6. The joint torques and positions were measured using the Barrett API.

In order to perceive auditory stimuli, the robot is equipped with two Audio-Technica U853AW microphones. They are mounted in the robot's head, directly below its eyes. One of the microphones is shown in Figure 3.4b. The output of the microphones is fed through two ART Tube MP Studio Microphone pre-amplifiers, which are both fed into a Lexicon Alpha bus-powered interface. The bus-powered interface is connected over a USB cable to the Linux PC that controls the robot. For the experiments described in this thesis, audio feedback was captured at the standard 16-bit/44.1 kHz over a single channel.

The robot is equipped with three cameras for processing visual data. Each of the robot's two eyes is an RGB Logitech Webcam, one of which is shown in Figure 3.4a. A Microsoft Kinect RGBD camera was attached to the metal stand supporting the robot during the experiments described in Chapter 6. The Kinect is shown in Figure 3.5. Vision was not as useful as

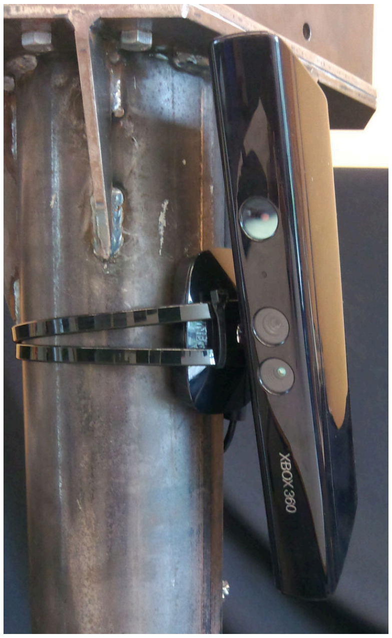


Figure 3.5: The robot's Microsoft Kinect RGBD camera. It was attached to the metal stand supporting the robot during the experiments described in Chapter 6.

audio and proprioception for solving several of the tasks, so while visual data was recorded in all three experiments, it was only used in the experiments described in Chapter 6. In those experiments, the robot recorded visual feedback only from the Kinect. The Kinect records RGB data at 640x480.It also records depth information for each pixel. Because all objects used in Chapter 6 have the same shape, only the RGB data was used. Some example images recorded during those experiments are shown in Figure 3.6.

3.3 The Robot's Behaviors

In all three experiments the robot used stereotyped exploratory behaviors to interact with the objects. Each behavior was defined as a series of waypoints in joint space. The default PID controller of the WAM API was used to move the arm between each of the waypoints. During the execution of each behavior on an object, the robot recorded from its sensory modalities. An example recording from an interaction is shown in Figure 3.6. The specific behaviors used in each experiment are described in more detail in the following chapters.

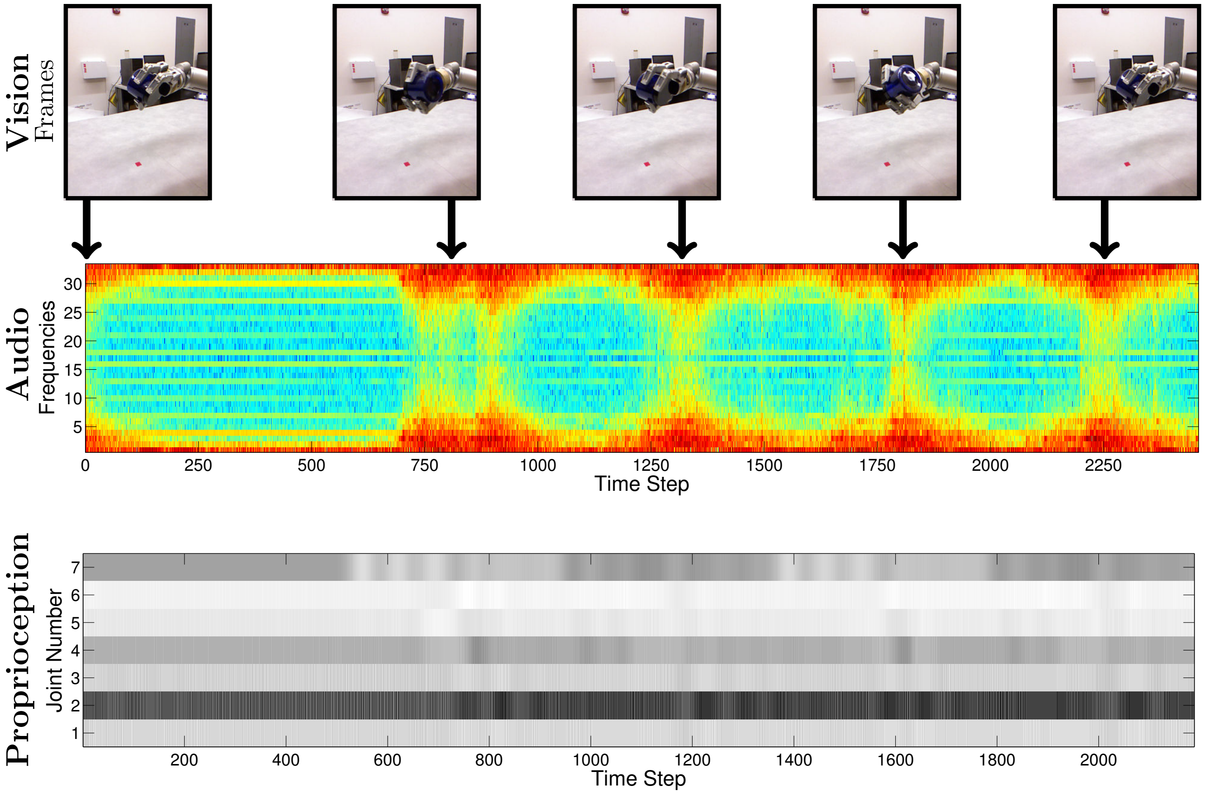


Figure 3.6: An example data stream recorded by the robot during one interaction with an object (the behavior in this case is *rattle*). The top row of the figure shows a selection of images recorded by the robot's Kinect camera. The middle row shows the audio spectrogram computed from the audio data recorded from the robot's microphones. The bottom row shows the torques applied to the robot's arm during the interaction.

CHAPTER 4. THE OBJECT PAIRING AND MATCHING TASK: TOWARD MONTESSORI TESTS FOR ROBOTS[*]

The Montessori method is a popular approach to education that emphasizes student-directed learning in a controlled environment. Object matching is one common task that children perform in Montessori classrooms. Matching tasks also occur quite frequently on intelligence tests for humans, which suggests that intelligence correlates with the skills required to solve these tasks. This chapter describes robotic experiments with four Montessori matching tasks: sound cylinders, sound boxes, weight cylinders, and pressure cylinders. The robot grounded its representation for the twelve objects in each task in terms of the auditory and proprioceptive outcomes that they produced in response to a set of ten exploratory behaviors. The results show that based on this representation, it is possible to identify task-relevant sensorimotor contexts (i.e., exploratory behavior and sensory modality combinations) that are useful for performing matching on a given set of objects. Furthermore, the results show that as the number of sensorimotor contexts used to perform matching increases, the robot's ability to match the objects also increases.

4.1 Introduction

The Montessori method is a 100-year-old method of schooling that was developed by Maria Montessori (1870-1952), an influential Italian educator. It emphasizes *embodied cognition*, requiring that children actively touch, move, relate, and compare objects (Lillard, 2008). The Montessori method focuses on student-directed learning activities with a specialized set of educational materials (Montessori, 1912; Lillard, 2008; Lillard and Else-Quest, 2006). It attempts

[*] This chapter is based on the paper: C. Schenck and A. Stoytchev, "The object pairing and matching task: Toward Montessori tests for robots," in Proceedings of the Workshop on Developmental Robotics, held at the 2012 IEEE-RAS International Conference on Humanoid Robotics, Osaka, Japan.

to stimulate the development of different skill sets, including sensory development, language development, and numeracy skills (Lillard, 2008).

One task typical for a Montessori classroom is object matching. Children are given two sets of objects and asked to find the matches from one set to another. Sample tasks include matching colored tiles, matching 3-dimensional shapes, and matching pieces of textured cloth (Pitamic, 2004). All these tasks are designed to stimulate a child's ability to perceive object properties and to allow the child to learn about the nature of objects and their similarities.

The skills required to perform matching are also useful for other tasks such as object grouping, category recognition, and object ordering. At a fundamental level, these skills require the ability to find differences between similar objects and similarities between different objects. Recent work in robotics has found that robots are able to recognize objects and their categories (Sinapov et al., 2011a; Saenko and Darrell, 2008), group objects in an unsupervised manner (Endres et al., 2009), and find the odd one out in a set of objects (Sinapov and Stoytchev, 2010b). These studies all strongly suggest that a robot should be able to solve object pairing tasks.

This chapter describes a method that allows a robot to identify and match object pairs within a set of objects based on their sensorimotor properties. To do this, the robot first interacted with the objects using a set of exploratory behaviors (*grasp, lift, hold, shake, rattle, drop, tap, poke, push,* and *press*) in order to ground the properties of the objects in the robot's behavioral repertoire. After interacting with the objects, the robot performed feature extraction on the raw sensory data to create sensory feedback sequences for each interaction. For each object, the robot recorded both proprioceptive feedback in the form of joint torques and auditory feedback in the form of an audio spectrogram. Next, the robot generated similarity scores for all possible object pairs and used these scores to match the objects. To combine information from different sensorimotor contexts (e.g., *audio-drop* and *proprioception-shake*), the robot used three different methods: uniform-weight combination, recognition accuracy based weight combination, and pairing accuracy based combination. These methods were evaluated for their ability to match standard Montessori objects.

This experiment used four typical Montessori matching tasks. In each task there were two

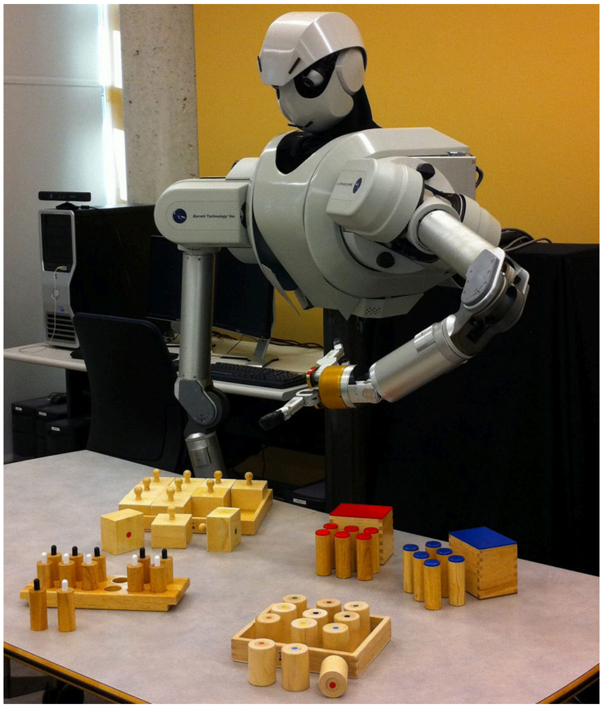


Figure 4.1: The robot and the four Montessori matching tasks that were used in the experiments. In clockwise order, the four tasks were: sound cylinders, weight cylinders, pressure cylinders, and sound boxes.

groups of six objects and the goal was to find the matching pairs of objects between the two groups. The results indicate that the estimated object similarities were sufficient to adequately pair objects. The robot was able to solve the object matching task with a high degree of accuracy. Furthermore, the robot was able to identify the functionally meaningful sensorimotor contexts in which it can distinguish between objects. To the best of our knowledge, this is the first experiment that has applied Montessori learning techniques in a robotic setting.

4.2 Experimental Setup

4.2.1 Robot and Sensors

The experiments in this study were performed with the upper-torso humanoid robot shown in Figure 4.1. The robot has as its actuators two 7-DOF Barrett Whole Arm Manipulators (WAMs), each with an attached Barrett Hand. Each WAM has built-in sensors that measure joint angles and torques at 500 Hz. An Audio-Technica U853AW cardioid microphone mounted

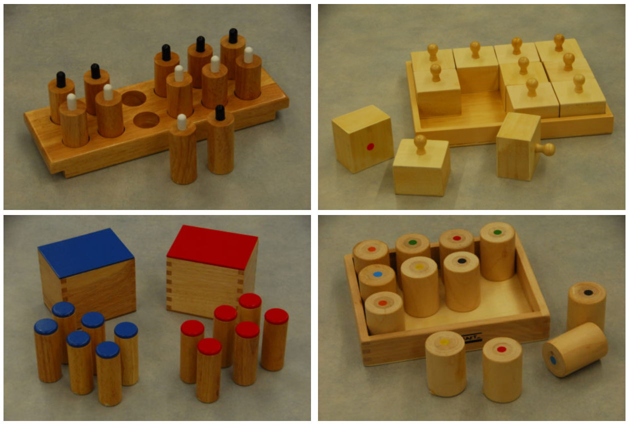


Figure 4.2: The four sets of Montessori objects used in the experiments. From left to right and top to bottom the object sets are: *pressure cylinders*, *sound boxes*, *sound cylinders*, and *weight cylinders*. All objects are marked with colored dots on the bottom to indicate the correct matches; other than that, the objects in each set are all visually identical (except for the *pressure cylinders* and the *sound cylinders*, which also have different colors for the tops to indicate the two sets of six objects).

in the robot's head was used to capture auditory feedback at the standard 16-bit/44.1 kHz over a single channel. Chapter 3 describes the robotic platform in more detail.

4.2.2 Objects

The robot explored four standard Montessori sets of objects: *pressure cylinders*, *sound boxes*, *sound cylinders*, and *weight cylinders* (Figure 4.2). Each set is composed of six pairs of objects. The objects in each pair are functionally identical to each other. The objects in each set are designed to vary in one specific dimension and be identical in all other dimensions. The *pressure cylinders* vary in the amount of force required to depress the rod, with pairs requiring the same amount of force. The *sound boxes* vary in the sounds they make when the contents move around inside the box, with pairs making the same sounds. The *sound cylinders* vary in the same way as the *sound boxes*, but are cylindrical in shape and have different contents than the boxes. The *weight cylinders* vary by weight, going from light to heavy, with pairs having the same weight.

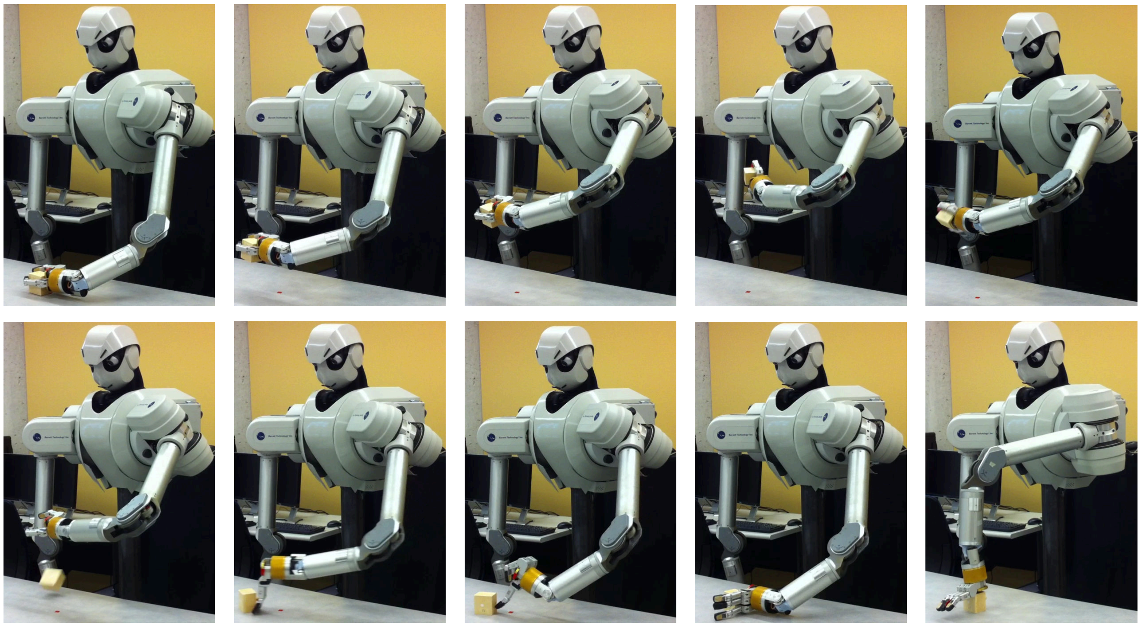


Figure 4.3: The ten exploratory behaviors that the robot performed on all objects. From left to right and top to bottom: *grasp*, *lift*, *hold*, *shake*, *rattle*, *drop*, *tap*, *poke*, *push*, and *press*. The object in this figure is one of the sound boxes. The red marker on the table indicates the initial position of the objects at the beginning of each trial. The object was placed back in that position by the experimenter after some of the behaviors (e.g., drop).

4.2.3 Exploratory Behaviors

The robot used ten behaviors to explore the objects: *grasp*, *lift*, *hold*, *shake*, *rattle*, *drop*, *tap*, *poke*, *push*, and *press*. All of these exploratory behaviors, except rattle, have been used in our previous work (Sinapov et al., 2013), i.e., they were not specifically designed for the Montessori objects used in this chapter. The behaviors were performed with the robot's left arm and encoded with the Barrett WAM API as trajectories in joint-space. The default PID controller of the WAM was used to execute the trajectories. Figure 4.3 shows images of the robot performing each behavior on one of the sound boxes. All exploratory behaviors were performed identically on each object, with only minor variations due to the initial placement of the objects by the experimenter.

4.2.4 Data Collection

The robot interacted with the objects by performing a series of exploration trials. During each trial, an object was placed at a marked location on the table by the experimenter and the robot performed all ten of its exploratory behaviors on the object. The experimenter then picked another object and the robot repeated this process. This was done until each object had been explored ten times. During each interaction, the robot recorded proprioceptive information in the form of joint torques applied to the arm and auditory data captured by the microphone. The robot also recorded visual data, but it was not used in this experiment. In the end, the robot performed all ten behaviors ten times on each of the twelve objects in the four sets, resulting in $10 \times 10 \times 12 \times 4 = 4800$ behavior executions. This resulted in 18 GB of data, which was stored for off-line analysis. It took approximately 20 hours to collect this dataset.

4.3 Feature Extraction

We used the method and the publicly available source code for proprioceptive and auditory feature extraction that is described by Sinapov et al. (2011a). It is briefly summarized below. Proprioceptive data was recorded as joint torques over time resulting in a $7 \times m$ matrix, in which each column represents one set of torque readings for all joints and m is the number of readings. To reduce noise, a moving-average filter was applied over each row in the matrix, which corresponds to the torques from one joint. Audio data was recorded as wave files, one for each interaction. A log-normalized Discrete Fourier Transform was performed on each audio file using $2^5 + 1 = 33$ frequency bins resulting in a $33 \times n$ matrix, where each column represents the activation values for different frequencies at a given point in time and n is the number of samples in the interaction. The Growing Hierarchical Self-Organizing Map (SOM) toolbox (Chan and Pampalk, 2002) was used to map each column to a single state. Two 6×6 SOMs were trained (one for audio and one for proprioception) using 5% of the columns that were randomly selected from all the joint torque and auditory data recorded by the robot. Each joint torque and auditory record was then mapped to a discrete sequence of states, where each column in the record was represented by the most highly activated SOM state for that column.

For more details see Sinapov et al. (2011a).

4.4 Experimental Methodology

4.4.1 Estimating Similarity

Given a set of objects $\mathcal{O}$ the robot must be able to estimate the pairwise similarity for any two objects $i, j \in \mathcal{O}$ in a given sensorimotor context (i.e., exploratory behavior and sensory modality combination). Let $\mathcal{X}_c^i = [X_1, ..., X_D]$ be the set of sensory feedback sequences detected while interacting with object $i \in \mathcal{O}$ in sensorimotor context $c \in \mathcal{C}$ (where $\mathcal{C}$ is the set of all contexts) and let $sim(X_a, X_b)$ be the similarity between two sequences X_a and X_b. The similarity between objects i and j can be approximated with the expected pairwise similarity of the sequences in $\mathcal{X}_c^i$ and $\mathcal{X}_c^j$:

$$s_{ij}^c = \mathbf{E}[sim(X_a, X_b) | X_a \in \mathcal{X}_c^i, X_b \in \mathcal{X}_c^j].$$

We used the Needleman-Wunsch global alignment algorithm (Navarro, 2001) to calculate $sim(X_a, X_b)$. The algorithm calculates the cost of aligning two discrete sequences (strings), which in our case correspond to sequences of most highly-activated SOM states (see the previous section). The expected similarity s_{ij}^c is estimated as

$$\frac{1}{|\mathcal{X}_c^i| \times |\mathcal{X}_c^j|} \sum_{X_a \in \mathcal{X}_c^i} \sum_{X_b \in \mathcal{X}_c^j} sim(X_a, X_b).$$

Next, the robot estimates the $|\mathcal{O}| \times |\mathcal{O}|$ pairwise object similarity matrix $\mathbf{W}^c$ for a specific sensorimotor context $c \in \mathcal{C}$. Each entry W_{ij}^c in $\mathbf{W}^c$ is defined as the similarity s_{ij}^c between two objects i and j in the specific context c. Figure 4.4 shows the similarity matrices for the *sound cylinders* for each of the 20 contexts.

4.4.2 Combining Sensorimotor Contexts

It has been shown that combining information from different sensorimotor contexts has a boosting effect for tasks such as object recognition (Sinapov and Stoytchev, 2010a). Since object matching is a similar task, it is likely that combining contexts will be useful in this case as

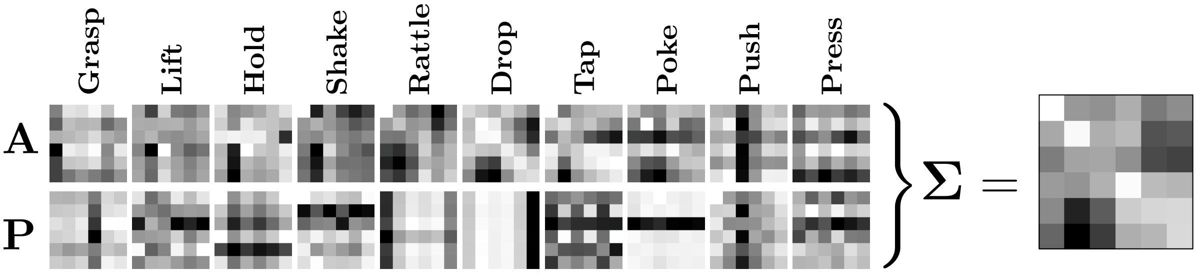


Figure 4.4: The similarity matrices used to perform matching given two sets of six objects each for the *sound cylinders*. The matrices for each individual context are shown as well as the consensus matrix for all 20 contexts ("**A**" denotes matrices computed from *audio* and "**P**" denotes matrices computed from *proprioception*). The pairing accuracy combination method using four pairs for training was used to combine the individual matrices. In each matrix lighter colors denote more similarity while darker colors denote more dissimilarity.

well. Thus, we propose three methods to combine sensorimotor contexts: uniform combination, recognition accuracy based combination, and pairing accuracy based combination. The result of combining different contexts is a consensus matrix **W** that represents the similarity between object pairs for the specific set of contexts that was used to create it.

4.4.2.1 Uniform Combination

Given some set of contexts $\mathcal{C}'$, where $\mathcal{C}' \subseteq \mathcal{C}$, the similarity matrices $\mathbf{W}^c$ for each of these contexts can be used to construct the consensus matrix **W** by simply averaging their individual values, i.e.,

$$W_{ij} = \frac{1}{|\mathcal{C}'|} \sum_{c \in \mathcal{C}'} W_{ij}^c$$

for all pairs of objects i and j.

4.4.2.2 Recognition Accuracy Based Combination

This method assumes that contexts that are useful for object recognition will also be useful for object pairing. The object recognition accuracy r_c for context c is estimated by performing 10-fold cross validation on all the data from context c using a classifier that attempts to recognize object identities from sensory feedback sequences. To create the consensus matrix

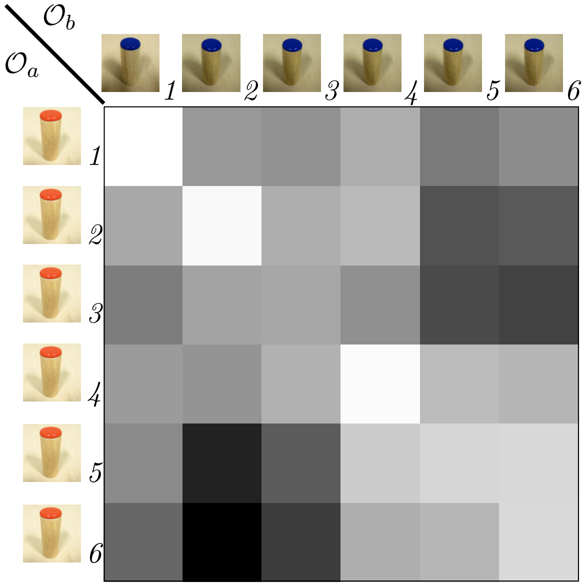


Figure 4.5: The consensus weight matrix for the *sound cylinders* using all 20 sensorimotor contexts for matching two groups of six objects. The pairing accuracy combination method using four pairs to train was used to combine the individual similarity matrices for each context. The subscripts indicate correct matches. This matrix is identical to the one shown in the right-hand side of Figure 4.4.

for a given set of contexts $\mathcal{C}'$ ($\mathcal{C}' \subseteq \mathcal{C}$), a weighted combination was used:

$$W_{ij} = \sum_{c \in \mathcal{C}'} \alpha_c \times W^c_{ij}$$

where α_c is the normalized recognition accuracy r_c for context c such that $\sum_{c \in \mathcal{C}'} \alpha_c = 1.0$. The classifier used was the k-nearest neighbor classifier with k set to 3 and using the global alignment similarity function as a similarity metric.

4.4.2.3 Pairing Accuracy Based Combination

The third combination method allowed the robot to get feedback on its attempts to pair some of the objects and to refine its ability to pair the remaining objects. In order to determine the usefulness of each context, the robot split the set of objects such that either 2, 3, or 4 of the six pairs were in the training set and the rest remained in the testing set. Then, for each context c, using the objects in the training set, the robot would attempt to pair them (using the pairing method described below) and evaluate the pairing accuracy p_c for that context. To construct the consensus matrix $\mathbf{W}$, a weighted combination was used similar to the previous method:

$$W_{ij} = \sum_{c \in \mathcal{C}'} \alpha_c \times W_{ij}^c$$

where α_c is the normalized pairing accuracy p_c for context c such that $\sum_{c \in \mathcal{C}'} \alpha_c = 1.0$. After generating the consensus matrix $\mathbf{W}$, the robot would then attempt to pair only the objects from the testing set. Figures 4.4 and 4.5 show a consensus matrix generated by combining the similarity matrices from all 20 contexts when training using 4 pairs of objects.

4.4.3 Generating Matchings

The robot was tasked with generating matchings among the objects in the four sets of Montessori toys. The objects in each set were split into two groups of six and the robot was tasked with selecting one object from each group to generate a match. This split into two groups of six is naturally suggested by the Montessori toys. For example, the sound cylinders have either red or blue caps; the pressure cylinders have either black or white buttons (see Figure 4.2).

More formally, given a 6x6 non-symmetric similarity matrix $\mathbf{W}^c$ or a consensus matrix $\mathbf{W}$ and objects $\mathcal{O}$ partitioned into two sets of equal size $\mathcal{O}_a$ and $\mathcal{O}_b$, matches were generated by picking pairs that maximized similarity between the objects in the pair and minimized similarity between those objects and the remaining objects. One such matrix is shown in Figure 4.4. Formally, the objects $i \in \mathcal{O}_a$ and $j \in \mathcal{O}_b$ that maximize

$$q(i, j, \mathbf{W}) = W_{ij} - \gamma \left[\sum_{k \in \mathcal{O}_b / j} W_{ik} + \sum_{k \in \mathcal{O}_a / i} W_{kj} \right]$$

were selected and then removed from $\mathcal{O}_a$ and $\mathcal{O}_b$. The first term captures the pairwise similarity between objects i and j; the last term captures the pairwise similarity between objects i and j and the rest of the objects. The constant γ is a normalizing weight, which ensures that this function is not biased toward any of the terms. In our case, it was set to

$$\gamma = \frac{1}{2(|\mathcal{O}| - 1)}.$$

This process was repeated until no more objects remained to be paired.

4.4.4 Evaluation

Given a set of objects (e.g., the weight cylinders), the robot's model was queried in order to group the objects into pairs. Five interactions were randomly picked for each object from the set of ten interactions that were performed on each object (recall that the robot performed the same behavior 10 times on each object) and used to create the weight matrix $\mathbf{W}^c$ for each sensorimotor context $c \in \mathcal{C}$. Consensus matrices $\mathbf{W}$ were generated using the three methods described above for a given set of contexts. Matchings were then generated using the method described above. This process was repeated 100 times for every group of contexts. For each size from 1 to $|\mathcal{C}|$, 100 sets of contexts were randomly generated and tested ($1,721$ in total)[1]. Results are reported as the average accuracy or as Cohen's kappa statistic (Cohen, 1960) over all 100 iterations.

Accuracy is computed as

$$\%Accuracy = \frac{\#correct\ matchings}{\#total\ matchings} \times 100.$$

The kappa statistic is computed as

$$kappa = \frac{P(a) - P(e)}{1 - P(e)}.$$

In our experiments, $P(a)$ is the pairing accuracy of the robot and $P(e)$ is the accuracy a random matching would be expected to get. Kappa is used to allow for direct comparisons between the different sensorimotor context combination methods, since for the pairing accuracy based method, chance accuracy is different than it is for the other methods. The kappa statistic controls for chance accuracy.

The evaluation was performed off-line after the robot interacted with all 48 objects (4 Montessori tasks × 12 objects in each).

[1] For sets of size 1, $|\mathcal{C}| - 1$, and $|\mathcal{C}|$ all sets of that size were tested since there were fewer than 100 sets of those sizes.

4.5 Results

4.5.1 Object Matching with a Single Context

Figure 4.6 shows the matching accuracy for each context for all four Montessori tasks. For the *pressure cylinders*, the best sensorimotor context was *proprioception-press* (97.5% pairing accuracy), which was expected. Surprisingly, *audio-press* also did well (80.7%), which was not expected since (at least to the authors' ears) all the cylinders sound the same when pressed. Also interesting is the *audio-drop* context for the *sound cylinders* (89.3% accuracy), which outperformed both *shake* (60.3%) and *rattle* (51.3%) behaviors for *audio*. *Audio-press* (82.3%) for the *sound cylinders* also did well, which is likely due to the fact that they would fall over while being pressed. It is also worth noting that for the *weight cylinders*, the best contexts were *proprioception-shake* (87.7%) and *proprioception-push* (94.3%) rather than contexts that more directly measure the weight such as *proprioception-lift* (50.7%) and *proprioception-hold* (18.8%).

In summary, the robot was able to identify the relevant behaviors and sensory modalities and use them to pair the objects in each of the four Montessori tasks with a high degree of accuracy.

4.5.2 Object Matching with Multiple Contexts

Figure 4.7 shows the kappa statistic for each set of objects as the number of contexts is varied from 1 to 20. The graphs show that as the number of sensorimotor contexts used to perform matching increases, so does the kappa statistic. In all cases, the pairing accuracy based combination using four pairs for training (the cyan line) outperforms all the other combination methods. The only exception to this is for the *sound boxes*, since accuracy reaches 100% and all methods reach a kappa value of 1.0. In most cases, the pairing accuracy based combination using three pairs for training (the yellow line) also outperforms the other methods (except for the method that uses four pairs for training). The pairing accuracy based combination using two pairs for training performs about the same as the recognition accuracy combination method, which usually performs slightly better than the uniform combination method. All

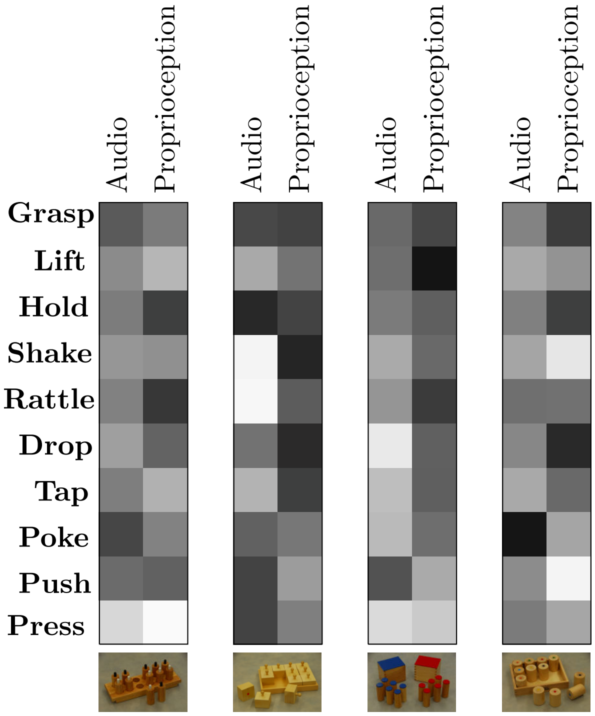
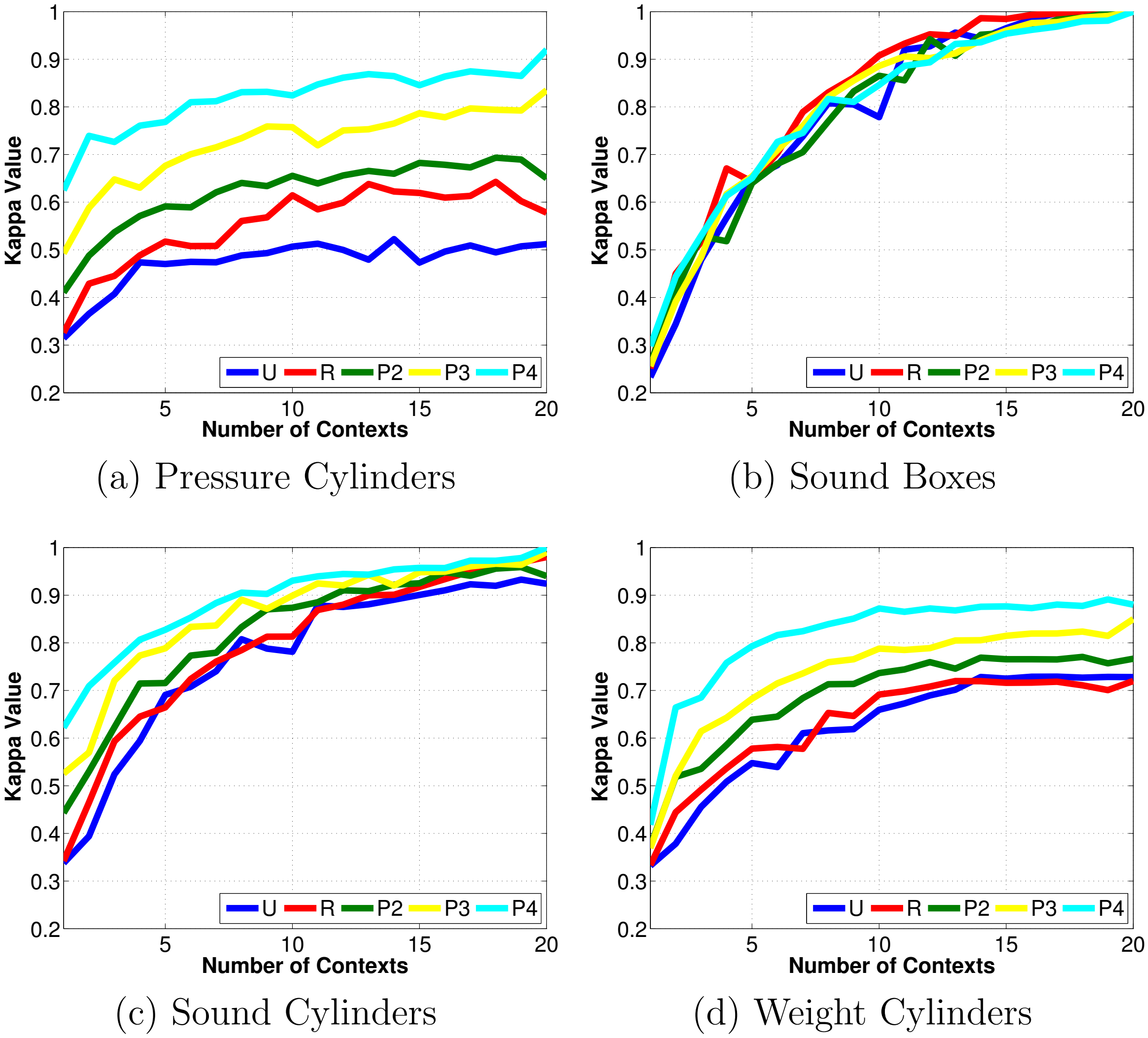


Figure 4.6: The accuracy of each context when matching between two sets of six objects. Lighter values indicate higher accuracy with completely white being 100%. Darker values indicate lower accuracy with completely black being 0%. The images from left to right are: *pressure cylinders*, *sound boxes*, *sound cylinders*, and *weight cylinders*.



(a) Pressure Cylinders

(b) Sound Boxes

(c) Sound Cylinders

(d) Weight Cylinders

Figure 4.7: The kappa statistic for each set of objects. Each line represents a different method for combing the sensorimotor contexts. The line labels are as follows: U-uniform combination; R-recognition accuracy based combination; P2-pairing accuracy using two pairs for training; P3-pairing accuracy using three pairs for training; P4-pairing accuracy using four pairs for training.

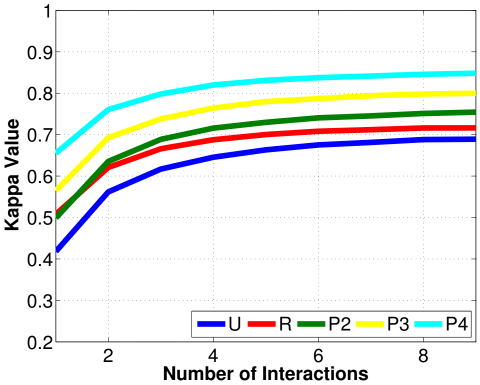


Figure 4.8: The kappa statistic averaged across all four sets of objects while varying the number of interactions used to generate the similarity matrices $\mathbf{W}^c$ for each context $c \in \mathcal{C}$. The number of randomly sampled interactions was varied from 1 to 9. The line labels are the same as in Figure 4.7.

combination methods perform better than chance for all object sets, which is indicated by a 0.0 kappa value.

4.5.3 Repeating the Same Behavior

In all results reported up to this point, five interactions were randomly chosen from the ten for each object during each iteration. Figure 4.8 shows the average kappa statistic as the number of interactions vary, averaged over all the sets of objects and number of contexts. The accuracies quickly converge after only a few trials, implying that repeating the same behavior multiple times on an object has quickly diminishing returns. In most cases and for all combination methods, after four repetitions there is very little gain. Diminishing returns is most quickly realized for the pairing accuracy combination method using four pairs for training. The largest gain when increasing interactions was realized by the uniform combination method. This suggest that the uniform combination method benefited the most from a decrease in noise due to its lack of weighted preferences between the contexts, whereas the pairing accuracy combination methods didn't benefit as much because the weights assigned to each context already decreased the noise.

4.6 Summary

This chapter demonstrated a framework that allows a robot to solve object matching tasks by estimating the pairwise similarity of objects in specific sensorimotor contexts. The performance of this framework was evaluated with four standard Montessori tasks that require pairing a set of objects based on their perceived similarities across multiple sensory modalities. The results showed that for a given set of objects, certain contexts are best suited to extract the information necessary to perform object pairing (e.g., *audio-shake* for the *sound boxes*), while others are not useful for that set of objects (e.g., *proprioception-lift* for the *sound cylinders*).

The robot was also able to combine similarity measures from different contexts using three different methods: uniform combination, recognition accuracy based combination, and pairing accuracy based combination. The robot was able to achieve the best performance in almost every case when it was allowed to train on four of the six object pairs before being tested on the remaining two. These results show that embodied sensorimotor similarity measures between objects can be extremely useful for performing matching tasks.

This chapter showed that embodied learning can be very useful for solving object pairing tasks. For each set of objects the robot learned which set of contexts are most useful for pairing the objects and which are not. The objects in each Montessori task implicitly capture an important concept that the robot can discover on its own through sensorimotor exploration. In the next chapters we will utilize this in order to solve other multi-object tasks. Chapter 5 uses two of the sets of objects from this experiment to analyze how robots can solve the order completion task. Chapter 6, while not directly using any Montessori objects, uses objects that are designed based on the same principles as Montessori objects.

CHAPTER 5. WHICH OBJECT COMES NEXT? GROUNDED ORDER COMPLETION BY A HUMANOID ROBOT[*]

This chapter describes a framework that a robot can use to complete the ordering of a set of objects. Given two sets of objects, an ordered set and an unordered set, the robot's task is to select one object from the unordered set that best completes the ordering in the ordered set. In our experiments, the robot interacted with each object using a set of exploratory behaviors, while recording feedback from two sensory modalities (audio and proprioception). For each behavior and modality combination, the robot used the feedback sequence to estimate the perceptual distance for every pair of objects. The estimated object distance features were subsequently used to solve ordering tasks. The framework was tested on object completion tasks in which the objects varied by weight, compliance, and height. The robot was able to solve all of these tasks with a high degree of accuracy.

5.1 Introduction

Humans can detect order in an unordered set of objects at a very early age. Ordering tasks frequently appear on modern intelligence tests (Hagmann-von Arx et al., 2008; Kaufman, 1994). They are also tightly integrated in many educational methodologies. For example, in the Montessori method (Montessori, 1912), a 100-year-old method of schooling for children that has been shown to outperform standard methods (Lillard, 2008; Lillard and Else-Quest, 2006), children are encouraged to solve different object ordering tasks with specialized toys (Pitamic, 2004). These strongly suggest that the ability to discover orderings among sets of objects is an

[*] This chapter is based on the paper: C. Schenck, J. Sinapov, and A. Stoytchev, “Which object comes next? Grounded order completion by a humanoid robot,” Journal of Cybernetics and Information Technologies, vol. 12, no. 3, pp. 5–16, 2012.

important skill. Indeed, studies in psychology have revealed that this skill is learned at a very early age (Sugarman, 1981; Graham et al., 1964; Ebeling and Gelman, 1988, 1994).

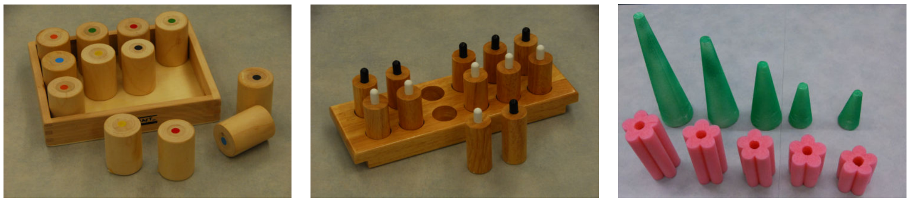
Because order completion skills are so important for humans they should be important for robots that operate in human environments as well. Previous research has shown that robots can successfully form object categories (Nolfi and Marocco, 2002; Natale et al., 2004; Nakamura et al., 2007; Takamuku et al., 2008) and solve the odd-one-out task (Sinapov and Stoytchev, 2010b). Object ordering tasks, however, have not received a lot of attention from the robotics community to date.

This chapter proposes a method for discovering orderings among groups of objects. The experiments were conducted with an upper-torso humanoid robot, which interacted with the set of objects using a set of stereotyped exploratory behaviors. The robot recorded both auditory and proprioceptive data during each interaction and then extracted features from the sensory records. Using the extracted features for each object, the robot was able to estimate a pairwise distance matrix between every pair of objects. Then given three objects that form an ordered set, the robot's model was queried to pick one object from another group of four to complete the ordering in the first set. The results show that the robot was able to pick the correct object that completes the ordering with a high degree of accuracy and that different exploratory behaviors and sensory modalities are required to capture different ordering concepts.

5.2 Experimental platform

All experiments were performed with the lab's upper-torso humanoid robot, which has two 7-DOF Barrett Whole Arm Manipulators (WAMs) as its actuators, each with an attached Barrett Hand. The robot captured proprioceptive information from the built-in sensors in the WAM that measure the angles and the torques applied to each joint at 500 Hz. The robot also captured audio data through an Audio-Technica U853AW cardioid microphone mounted in its head at the standard 16-bit/44.1 kHz over a single channel. Chapter 3 describes the robotic platform in more detail.

The robot was tested on three ordering concepts: ordering by *weight*, ordering by *compliance*, and ordering by *height*. Figure 5.1 shows the three sets of objects that were used in the



(a) Weight Cylinders (b) Pressure Cylinders (c) Cones and Noodles

Figure 5.1: The three sets of objects used in the experiments

experiments. The first two are standard Montessori toys. The *weight cylinders* are composed of six pairs of objects (for a total of twelve objects) that vary by weight, with the objects in each pair having the same weight. All the *weight cylinders* are functionally identical except for their weight. The *pressure cylinders* are composed in a similar manner (six pairs of objects) except that they vary by the amount of pressure required to depress the rod on top of the object. The *cones* and *noodles* are composed of five green, styrofoam cones of varying sizes and five pink, foam pieces (cut from a water noodle) ranging in size from small to large. Because the object's in the first two sets are visually identical, this task cannot be solved with vision alone. In fact, the robot did not use vision at all to solve the ordering task.

The robot performed nine behaviors on each of the objects: *grasp*, *lift*, *hold*, *shake*, *drop*, *tap*, *poke*, *push*, and *press*. Additionally, the behavior *rattle* was performed on the *weight cylinders* and the *pressure cylinders*. Each behavior was encoded as a trajectory in joint-space for the left arm using the Barrett WAM API and executed using the default PID controller. All behaviors were performed identically on each object with the exception of *grasp* and *tap*, which were adjusted automatically based on the current visually detected location of the object. Figure 5.2 shows the robot performing each behavior on one of the pressure cylinders.

At the start of each trial, the experimenter placed one of the objects on the table in front of the robot. The robot then performed the exploratory behaviors on the object, with the experimenter placing the object back on the table if it fell off. This was repeated five times for each of the *cones* and *noodles* and ten times for the rest of the objects. The data for the *cones* and *noodles* was collected at an earlier time than for the rest of the objects, which is why only five repetitions were done and the behavior *rattle* was not performed on them. During

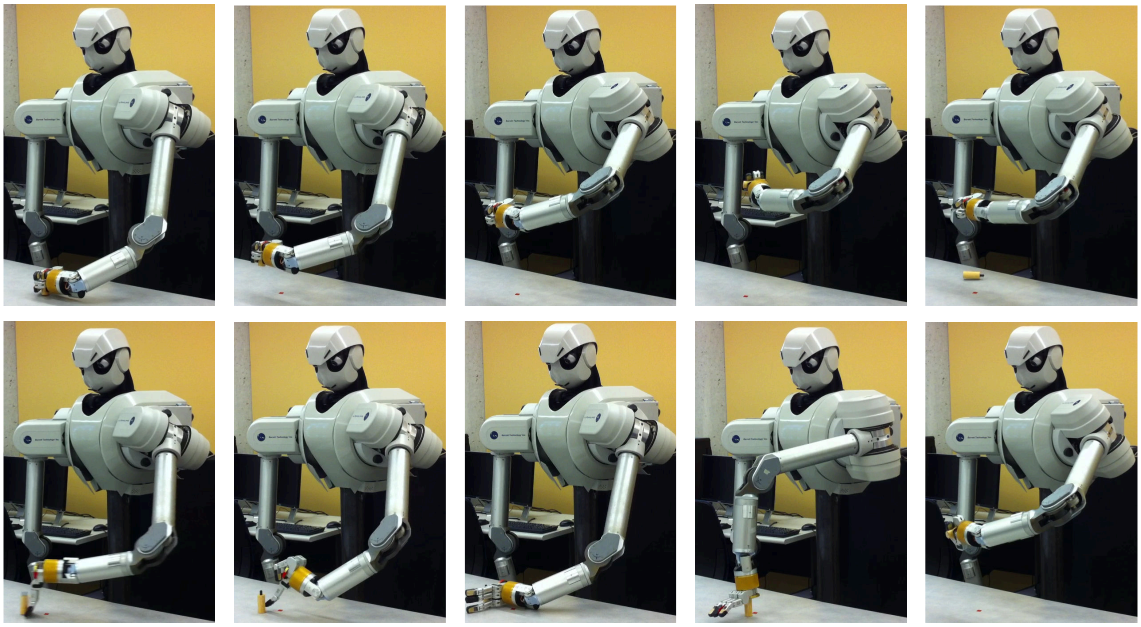


Figure 5.2: The ten exploratory behaviors that the robot performed on the objects. From left to right and top to bottom: *grasp*, *lift*, *hold*, *shake*, *drop*, *tap*, *poke*, *push*, *press*, and *rattle*. The *rattle* behavior wasn't performed on the *cones* and *noodles*. The object in this figure is one of the pressure cylinders. After some of the behaviors (e.g., drop), the object was moved back to the red marker location on the table by the experimenter.

each behavior, the robot recorded proprioceptive data in the form of joint torques applied to the arm over time and auditory data in the form of a wave file. Visual input was used only to determine the location of the object for the *grasp* and *tap* behaviors.

5.3 Feature extraction

5.3.1 Sensorimotor feature extraction

The auditory feedback from each behavior was represented as the Discrete Fourier Transform (DFT) of the sound's waveform, computed using 33 frequency bins. Thus, each interaction produced a $33 \times n$ matrix, where each column represented the intensities for different frequencies at a given point in time (i.e., n was the number of samples). The DFT matrix was further discretized uniformly into 10 temporal bins and 10 frequency bins. Thus, the auditory feature vector for each interaction was a $10 \times 10 = 100$ dimensional real-valued vector.

The proprioceptive feedback was represented as 7 time series of detected joint-torques, one for each of the robot's joints. To reduce the dimensionality of the data, each of the series was uniformly discretized into 10 temporal bins. Thus, the proprioceptive features for each interaction were represented by a $7 \times 10 = 70$ dimensional real-valued vector. As described next, the computed auditory and proprioceptive features were used to estimate the pairwise distances for each pair of objects.

5.3.2 Object feature extraction

Let $\mathcal{C}$ be the set of sensorimotor contexts, i.e., each $c \in \mathcal{C}$ corresponds to a behavior-modality combination (e.g., *audio-shake*), and let $\mathcal{O}$ denote the full set of objects. The goal of the object feature extraction routine is to compute a distance matrix $\mathbf{W}^c$ such that each entry $W^c_{ij} \in \mathbb{R}$ encodes how perceptually different objects o_i and o_j are in sensorimotor context c. Let the set $\mathcal{X}^c_i = [x_1, ..., x_D]^c_i$ contain the sensorimotor feature vectors detected for each of the D exploratory trials with object o_i in context c. The distance between two objects o_i and o_j in context c can be represented by the expected distance between the feature vectors in $\mathcal{X}^c_i$ and the feature vectors in $\mathcal{X}^c_j$, i.e.,

$$W^c_{ij} = \mathbf{E}[d_{L2}(x_a, x_b) | x_a \in \mathcal{X}^c_i, x_b \in \mathcal{X}^c_j],$$

where d_{L2} is the L2-norm distance function. This expectation is estimated by:

$$W^c_{ij} = \frac{1}{|\mathcal{X}^i_c| \times |\mathcal{X}^j_c|} \sum_{x_a \in \mathcal{X}^i_c} \sum_{x_b \in \mathcal{X}^j_c} d_{L2}(x_a, x_b).$$

The result is a set $\mathcal{W}$ of object distance matrices, where each $\mathbf{W}^c \in \mathcal{W}$ encodes the pairwise perceptual distance for each pair of objects in $\mathcal{O}$. The next section describes how these matrices can be used to decide which one of a given set of objects best completes a given order.

5.4 Methodology

5.4.1 Problem formulation

Each order completion task is formulated as follows. Let $\mathcal{O}$ denote the set of objects explored by the robot. Let $\mathcal{L}$ denote an *ordered subset* of $\mathcal{O}$, i.e., $\mathcal{L} = o_1, o_2, \ldots, o_N$ where each $o_i \in \mathcal{O}$.

Furthermore, let $\mathcal{G} \subset \mathcal{O}$ be an unordered set of M objects denoting the set of candidate objects that could be selected to complete the order. Finally, let $\mathcal{W}$ be a set of distance matrices such that for a given sensorimotor context c, the $|\mathcal{O}| \times |\mathcal{O}|$ matrix $\mathbf{W}^c \in \mathcal{W}$ encodes the pairwise object distances in that context.

In this setting, the task of the robot's model is to select one object from $\mathcal{G}$ that correctly completes the order specified by the ordered set $\mathcal{L}$. The idea behind the approach presented here is to define an objective function that can evaluate the quality of a proposed order and use that function to select an object from the set $\mathcal{G}$. The next sub-section describes the objective function as well as how that function is used to pick an object that completes the order.

5.4.2 Selecting the best order completion candidate

Let $q(\mathcal{L}, \mathbf{W}^c)$ denote the objective function that measures the quality of the order $\mathcal{L}$ with respect to the matrix $\mathbf{W}^c$. That function is defined as:

$$q(\mathcal{L}, \mathbf{W}^c) = \sum_{o_i \in \mathcal{L}} \sum_{o_j \in \mathcal{L}} \left(W^c_{ij} - d(o_i, o_j, \mathcal{L}) \right)^2 ,$$

where the function d is defined as

$$d(o_i, o_j, \mathcal{L}) = \sum_{r = o_i \ldots o_{(j-1)} \in \mathcal{L}} W^c_{r(r+1)}.$$

In other words, the function d approximates the distance between objects o_i and o_j by summing up the distances between adjacent elements in the ordered set $\mathcal{L}$. Thus, the function q measures the squared difference between the true distance matrix and the one approximated by the proposed ordering. It is used by the robot's model to complete a given ordered set of objects as follows. For each object o_k from the unordered set $\mathcal{G}$, let $\{\mathcal{L}, o_k\}$ denote the *ordered set* of objects produced by adding object o_k to the end of the ordered set $\mathcal{L}$. In this setting, the model selects the object o_k that minimizes the objective function $q(\{\mathcal{L}, o_k\}, \mathbf{W}^c)$.

5.4.3 Order completion using multiple sensorimotor contexts

The method presented so far can only use one distance matrix $\mathbf{W}^c$ that is specific to one sensorimotor context c. For many tasks, however, it may be desirable to use multiple sources of information about how objects relate to each other. For example, if the given ordered set of objects $\mathcal{L}$ is ordered by weight, there may be several exploratory behaviors that capture relevant proprioceptive information for solving the task (e.g., *lifting* and *holding* in place).

The set $\mathcal{W}$ contains multiple matrices encoding the pairwise object dissimilarities computed for a given set of sensorimotor contexts. For each object $o_k \in \mathcal{G}$, let the function completes$(\mathcal{L}, o_k, \mathbf{W}^c)$ return 1 if o_k is selected as the object completing the order and 0 otherwise. Given the set of all matrices $\mathcal{W}$, the ordered set $\mathcal{L}$, and the candidate set $\mathcal{G}$, the model selects the object $o_k \in \mathcal{G}$ that maximizes the following function:

$$\text{score}(o_k) = \sum_{\mathbf{W}^c \in \mathcal{W}} w_c \times \text{completes}(\mathcal{L}, o_k, \mathbf{W}^c),$$

where w_c is a weight that encodes the relevance of sensorimotor context c.

In the experiments described in the next section, three weighting methods are evaluated. Whereas everything in this chapter so far has been unsupervised, two of these weighting methods are supervised (methods 2 and 3). In the first method, the weights are *uniform.* In other words, for all c, $w_c = 1.0$.

In the second method, the weights are set to the estimated accuracy of using sensorimotor context c to solve the specific ordering task. In other words, the robot's model estimates the accuracy of a context c by running the method described in the previous subsection on a training set of tasks of the form $[\mathcal{L}, \mathcal{G}]$ for which the correct answers are known. Once the weights for all contexts have been estimated, the model uses those weights on subsequent tasks for which the answers are not known in advance.

The third method that was used to combine sensorimotor contexts is boosting. It was implemented using the AdaBoost algorithm (Freund and Schapire, 1995). It is briefly summarized here. Given a set of m tasks $[\mathcal{L}_1, \mathcal{G}_1], [\mathcal{L}_2, \mathcal{G}_2], ..., [\mathcal{L}_m, \mathcal{G}_m]$ for which the correct answers $o_k^1 \in \mathcal{G}_1, o_k^2 \in \mathcal{G}_2, ..., o_k^m \in \mathcal{G}_m$ are known, initialize the training weights as $D_1(i) = \frac{1}{m}$ for

$i = 1, ..., m$. For each iteration $t = 1, ..., T$, select the sensorimotor context $c^*(t)$ such that $c^*(t) = \arg\min_{c \in \mathcal{C}} \ \xi_c$. The error ξ_c of a context c is computed as

$$\xi_c = \sum_{i=1}^{m} D_t(i) \left[1 - \text{completes}(\mathcal{L}_i, o_k^i, \mathbf{W}^c)\right] ,$$

where $o_k^i \in \mathcal{G}_i$ is the object that correctly completes the ordering $\mathcal{L}_i$. Next, the parameter α_t is computed as a function of $\xi_{c^*(t)}$ as follows

$$\alpha_t = \frac{1}{2} \ln \frac{1 - \xi_{c^*(t)}}{\xi_{c^*(t)}},$$

where $\xi_{c^*(t)}$ is the error of the selected context in iteration t. After each iteration, the training weights for all $i = 1, ..., m$ are updated as follows

$$D_{t+1}(i) = D_t(i) \exp\left[-\alpha_t(2 * \text{completes}(\mathcal{L}_i, o_k^i, \mathbf{W}^{c^*(t)}) - 1)\right] ,$$

where $\mathbf{W}^{c^*(t)}$ is the object distance matrix of the context selected during iteration t, and then normalized such that they sum to 1. It is worthwhile to note that the following expression $-\alpha_t(2 * \text{completes}(\mathcal{L}_i, o_k^i, \mathbf{W}^{c^*(t)}) - 1)$ evaluates to $+\alpha_t$ if context $c^*(t)$ incorrectly predicts the object to complete the ordering $\mathcal{L}_i$ and $-\alpha_t$ otherwise. In essence, the training weights are altered such that tasks that context $c^*(t)$ is incorrect on are weighted higher and tasks that it is correct on are weighted lower.

Finally, the weight w_c for each sensorimotor context is computed by

$$w_c = \sum_{t=1}^{T} \alpha_t [c \equiv c^*(t)],$$

where $[c \equiv c^*(t)]$ is 1 if c was chosen during iteration t and 0 otherwise. In the experiments described in this chapter, T was set to 50. Results did not change significantly with a higher value for T.

5.4.4 Evaluation

The model was evaluated independently on each of the three ordering concepts. Fifty tasks were randomly sampled for each concept as follows: four objects were sampled from the set $\mathcal{O}$

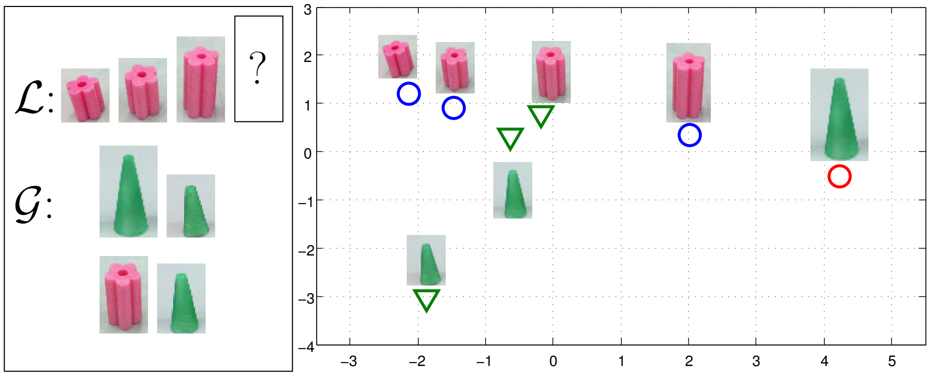


Figure 5.3: An example task. The box on the left shows both the ordered set $\mathcal{L}$ and the unordered set of objects $\mathcal{G}$ to choose from. The plot on the right shows the ISOMAP embedding of the distance matrix between the objects. The blue circles denote the three objects in $\mathcal{L}$, the red circle denotes the object in $\mathcal{G}$ that is selected to complete the order.

such that there existed a clear ordering amongst them (e.g., for the *weight cylinders*, two objects from the same pair would not be sampled together). The objects were then ordered (with the direction, forward or backward, determined randomly) and the last object was removed. Thus, the first three ordered objects formed the ordered set $\mathcal{L}$ for the given task. Three more objects were randomly sampled from the remaining objects in $\mathcal{O}$ such that none validly completed the ordering. These three objects, combined with the removed object, formed the set $\mathcal{G}$. The performance of the robot's model was evaluated in terms of accuracy, i.e., the number of tasks for which the robot's model picked the correct object divided by the total number of tasks.

For each concept, the performance of each sensorimotor context was evaluated. The accuracy was also computed as more and more contexts were used by the model. To estimate the context weights and to train the boosting method, five-fold cross-validation was performed with the 50 sampled tasks (i.e., 10 tasks were randomly assigned to each fold). The model was also evaluated as the number of tasks used for training was varied from 1 to 49. In this case, all contexts were used.

5.5 Results

5.5.1 An example order completion task

Figure 5.3 shows an example task in which the robot's model is tasked with completing an order of three objects that are ordered by height. In this case, the ordered input set, $\mathcal{L}$, consists of three pink noodles, while the candidate set, $\mathcal{G}$, contains four objects – three cones and one noodle, such that only one of them is taller than the last element in $\mathcal{L}$. In this specific case, the input distance matrix encoded the perceptual similarity of the objects in the *press-proprioception* sensorimotor context. The figure shows an ISOMAP (Tenenbaum et al., 2000) embedding of the distance matrix, which makes it easy to see that the matrix encodes an order between the objects. For this task, the model correctly picked the cone from the set $\mathcal{G}$ that is taller than the tallest noodle object in $\mathcal{L}$. The next subsection describes a quantitative evaluation of the model in which each sensorimotor context is evaluated on each of the three ordering tasks.

5.5.2 Ordering objects using a single sensorimotor context

For the first experiments, the performance of the model was evaluated using a single sensorimotor context. Figure 5.4 shows the accuracy for each context on each of the 3 concepts. As expected, *lift* (100%), *drop* (100%), *hold* (98.0%), *shake* (100%), and *rattle* (98.0%) for *proprioception* perform very well on the task of ordering objects by weight. This is likely because the robot was supporting the full weight of the object with its arm while performing these behaviors. For the *pressure cylinders*, *proprioception-lift* (100%) and *proprioception-tap* (98.0%) achieve high performance. The reason for this is likely due to the weight and moment of inertia differences in the objects caused by the different springs inside the *pressure cylinders*. *Proprioception-press* was able to achieve 100% accuracy on the *cones* and *noodles* task as was expected since the moment at which the arm touched the object varied depending on the object's height. The other sensorimotor contexts did not perform as well, with *proprioception-push* (84.0%) being the next highest performing context.

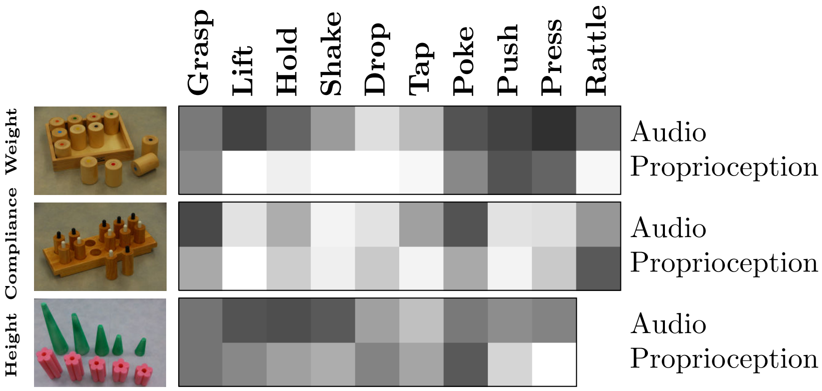
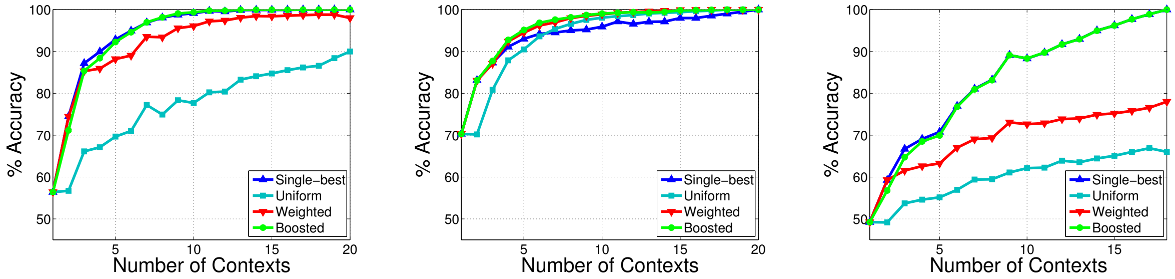


Figure 5.4: The accuracy of each context for each of the 3 concepts. Darker values indicate lower accuracies with solid black being 0%; lighter values indicate higher accuracies with solid white being 100%.

5.5.3 Ordering objects using multiple sensorimotor contexts

Figure 5.5 shows the performance of the robot on each concept as the number of contexts varies from 1 to $|\mathcal{C}|$ when using the uniform, weighted, and boosted combination methods as described in Section 5.4.3. The accuracy when picking just the single-best context (based on the training tasks) is also shown for comparison. As the number of combined contexts increases, the average accuracy also increases, which is consistent with our previous results (Sinapov et al., 2011a). Additionally, in every case the weighted combination method outperforms the uniform combination method. Also the boosted method always does at least as well as the weighted method and in most cases outperforms it. For the *weight cylinders* (Figure 5.5a), the weighted method reaches 98.0% accuracy and the boosted method reaches 100% accuracy when all contexts are used. For the *pressure cylinders* (Figure 5.5b) the weighted and boosted combination methods reach 100% when all contexts are used. For the *cones* and *noodles* (Figure 5.5c), the single-best context is able to achieve 100%, but when all the sensorimotor contexts are combined the robot was only able to achieve 72.0% accuracy using the weighted method. Using the boosted method, however, it was able to reach 100%. We believe that since for the *height* concept, unlike for the other two, there was only one context that performed well, the noise from combining underperforming contexts outweighed the single best performing context for the weighted method, but the boosted method was able to learn this and weight

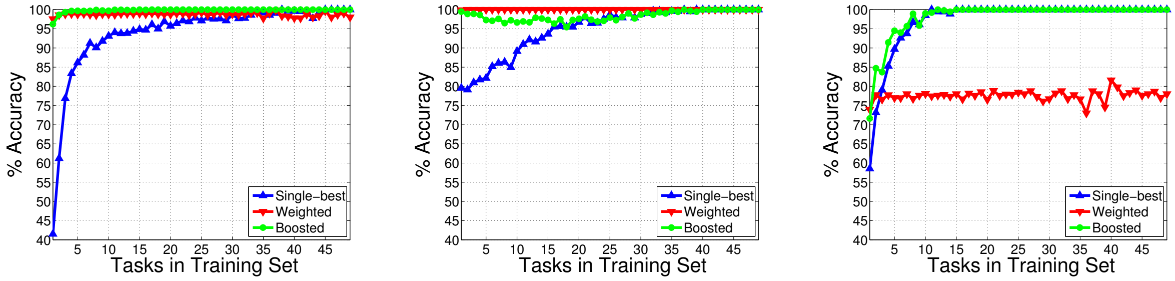


(a) Ordering by Weight (b) Ordering by Compliance (c) Ordering by Height

Figure 5.5: The accuracy as the number of contexts is increased. The blue line is the accuracy when picking the single-best context; the cyan line is the accuracy when using uniform weights to combine contexts; the red line is the accuracy when the contexts are weighted in proportion to their individual accuracies; and the green line is the accuracy achieved when using AdaBoost to learn the weights.

the best context higher.

Figure 5.6 shows the average accuracy as the number of tasks used for training is varied from 1 to 49 when combining all sensorimotor contexts. Again the single-best context (based on the training tasks) is shown for comparison. In every case, the weighted method converges after no more than 6 training tasks are used to estimate the weights. The boosted method always achieves 90% accuracy after no more than 4 training tasks and 95% accuracy after no more than 7. For *weight*, the boosted method and the weighted method converge at approximately the same rate. For *height*, the boosted method outperforms the weighted method by a large margin. For *compliance*, the boosted method converges slower than the weighted method (weighted reaches 100% after 3 tasks are used while boosted doesn't reach 100% until 40 tasks are used). This is likely related to the result in Figure 5.5, where *compliance* is the only task in which the uniform combination method reaches 100%. Interestingly, while the boosted method and the single-best context (as determined by the training set) converge to 100% for all three concepts, the boosted method converges much quicker for both the *weight cylinders* and *pressure cylinders*, and at about the same rate for the *cones* and *noodles*.



(a) Ordering by Weight (b) Ordering by Compliance (c) Ordering by Height

Figure 5.6: The average accuracy as the number of tasks used for training is increased. The blue line is the accuracy achieved when picking just the single-best context; the red line is the accuracy achieved when the contexts are weighted in proportion to their individual accuracies; and the green line is the accuracy achieved when using boosting. The results are averaged over 50 sets of training tasks for each size from 1 to 49.

5.6 Summary

This chapter presented a theoretical model for performing order completion. This model was evaluated using an upper-torso humanoid robot on three concepts: *weight*, *compliance*, and *height*. The results show that the robot was able to select objects to complete orderings with a high degree of accuracy. For each concept, there exists at least one sensorimotor context that was able to achieve 100% accuracy, and there were multiple such contexts for *weight* and *compliance*. When combining sensorimotor contexts, on average, the best performance was achieved when all contexts were used, though in every case the best single context did at least as well or better. This suggests that when completing an ordering determined predominantly by only one property (e.g., weight), if there exists at least one sensorimotor context that is able to capture that property, then its predictions will typically align with the true ordering.

Given these results, what strategy should the robot use to solve a novel order completion task? The results clearly show that the boosted combination method is the best strategy for combining sensorimotor contexts because it always performs as well as or better than every other method and because it usually takes very few training tasks to train. The methodology used in this chapter builds on our previous work, in which we have shown that stereotyped exploratory behaviors can be used to detect functional similarities between tools (Sinapov and

Stoytchev, 2008), perform object recognition (Sinapov et al., 2011a), perform object categorization (Sinapov and Stoytchev, 2011), recognize surface textures (Sinapov et al., 2011b), solve the odd-one-out task (Sinapov and Stoytchev, 2010b), and now solve the order completion task. These results suggest that a wide variety of tasks can be solved using a library of task-specific algorithms applied on a common set of sensorimotor data extracted from exploratory behaviors.

A limitation of the method described in this chapter is that while it can solve order completion tasks in which the order is ascending or descending by one property, it cannot solve tasks that require the synthesis of multiple properties. This is addressed in the next chapter, which presents a modified version of the methodology presented here that allows the robot to solve tasks that require the synthesis of multiple properties. The robot's ability to solve these more complicated tasks is tested using the matrix completion task.

CHAPTER 6. WHICH OBJECT FITS BEST? SOLVING MATRIX COMPLETION TASKS WITH A HUMANOID ROBOT

Matrix completion tasks commonly appear on intelligence tests. Each task consists of a grid of objects, with one missing, and a set of candidate objects. The job of the test taker is to pick the candidate object that best fits in the empty square in the grid. In this chapter we explore methods for a robot to solve matrix completion tasks that are posed using real objects instead of pictures of objects. Using several different ways to measure distances between objects, the robot detected patterns in each task and used them to select the best candidate object. When using all the information gathered from all sensory modalities and behaviors, and when using the best method for measuring the perceptual distances between the objects, the robot was able to achieve 99.4% accuracy over the posed tasks. This shows that the general framework described in this thesis is useful for solving matrix completion tasks.

6.1 Introduction

Intelligence tests have long been used to measure the Intelligence Quotient (IQ) of humans. One of the common types of problems on intelligence tests are matrix completion tasks. These problems consist of a grid of (usually) images, where one entry in the grid is missing. The job of the test taker is to select the object from a set of given candidates that best fits in the empty slot in the grid. The most well-known intelligence test that employs matrix completion tasks, the Raven's Progressive Matrices (RPM) test (Raven, 1938), has been shown to correlate strongly with the ability to understand the structure of complex environments (Raven, 2000). Other common intelligence tests, such as the Wechsler Abbreviated Scale of Intelligence (WASI)

(Wechsler, 1997), also have sections dedicated to matrix completions tasks.

Matrix completion tasks emphasize the ability to reason about the relationships between objects in the matrix, rather than merely recalling stored knowledge about the objects. In fact, John Raven developed the RPM test specifically to remove biases he saw in previous tests that made it difficult to accurately compare the scores of participants with and without extensive knowledge of concepts such as language (Watt, 1998). Currently there are no robotic systems that are capable of building longitudinal knowledge bases or understanding language to the same extent a human can. However, because matrix completion tasks do not require extensive background knowledge to solve, it is feasible to solve them with the current state of the art in robotics.

The concepts underlying matrix completion tasks frequently appear in places outside of intelligence tests as well. For example, the grid layout of the periodic table of the elements makes it easy to see the analogous relationships between elements in the same relative positions (e.g., cadmium can replace zinc in many vital enzymes, a relationship that is made apparent by the arrangement of the periodic table) (Scerri, 2011). In fact, when Mendeleev created the periodic table, due to the underlying properties of its layout, he was able to successfully predict the existence and properties of many yet undiscovered elements (Scerri, 2011). This suggests that the concepts underlying matrix completion tasks are very useful for solving other, related tasks in real-world environments.

This chapter uses the framework formulated in Chapter 1 to solve matrix completion tasks with a robot. The robot first interacted with a set of objects while recording from its auditory, visual, and proprioceptive sensory modalities. Then we randomly generated 500 matrix completion tasks using objects from the set and posed them to the robot. The robot generated a set of distance functions using four different methods: raw context distances (unsupervised and supervised), and category-based distances (unsupervised and supervised) (described in section 6.4.5). It then used them to attempt to deduce the patterns in the objects in the given matrix in order to select the best candidate object for each task. Using the best distance method, the robot was able to achieve 99.4% accuracy on the tasks. To the best of our knowledge, this is the first attempt at using a robot to solve matrix completion tasks.

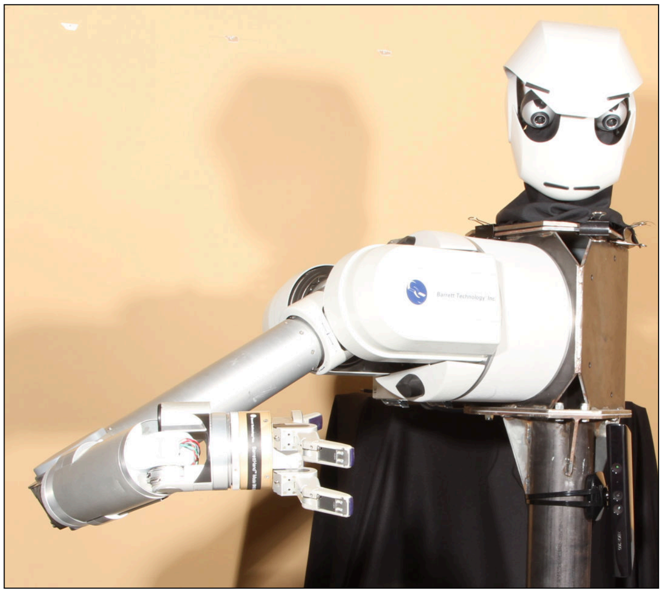


Figure 6.1: The robot used in these experiments. It is shown here with only its right arm as the left arm was temporarily removed for maintenance when these experiments were performed. The Microsoft Kinect camera is mounted on the lower part of the robot's torso.

6.2 Experimental Platform

6.2.1 Robot and Sensors

All experiments described in this chapter were performed using the robot shown in Figure 6.1. The robot is equipped with two 7-DOF Barrett Whole Arm Manipulators (WAMs), each with an attached Barrett Hand. Each WAM can measure its own joint angles and torques at a rate of 500 Hz. The robot used only its right arm to perform the behaviors in these experiments, as its left arm was temporarily removed for maintenance. The robot also has an Audio-Technica U853AW cardioid microphone mounted in its head in order to capture auditory feedback at the standard 16-bit/44.1kHz over a single channel. During the experiments, the robot was also equipped with a Microsoft Kinect camera, which can capture both RGB video and depth information. The Kinect camera was attached to the lower part of the robot's torso, slightly above the table and pointed down at it. The robot is described in more detail in Chapter 3.

6.2.2 Objects

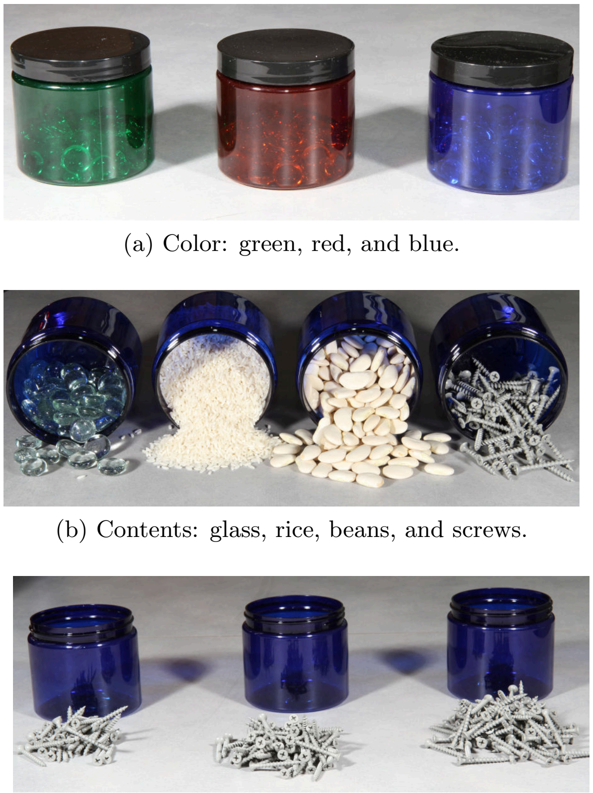
The objects used in the experiments described in this chapter were designed specifically to maintain the general structure of matrix completion tasks as described by Carpenter et al. (1990) while moving to the domain of physical objects (as opposed to images on a piece of paper). Figure 6.2 shows the three properties that the objects varied by. Each object is a cylindrical plastic jar that is 8.6 centimeters tall and 9.4 centimeters in diameter. The jars are semi-transparent, each being one of three colors: *blue*, *green*, or *red* (see Figure 6.2a). Each jar is filled with one of four different types of contents: *glass beads*, *rice*, *beans*, or *screws* (shown in Figure 6.2b). Each jar was filled until it weighed either 166g, 250g, or 337g (shown in Figure 6.2c). In all, there are 3 colors × 4 contents types × 3 weights = 36 total jars (one for each permutation of the values). Figure 6.3 shows all 36 objects.

6.2.3 Exploratory Behaviors

The robot performed ten stereotyped behaviors to explore the objects: *grasp*, *lift*, *hold*, *shake*, *rattle*, *drop*, *tap*, *poke*, *push*, and *press*. All of these behaviors are shown in Figure 6.4. In addition to these behaviors, the robot also performed the *look* behavior (not shown in Figure 6.4), during which it took a visual snapshot of the object on the table in front of it with the Kinect camera before performing the other behaviors on it. All behaviors in the experiments described in this chapter were performed with the robot's right arm and encoded using Barrett's API. The trajectory of the joint positions for each of the behaviors was executed using the default PID controller of the WAM. All of the behaviors were performed identically on each object, with only minor variations due to the initial placement of the object.

6.2.4 Sensorimotor Contexts

In this chapter, the robot used 21 sensorimotor contexts. A sensorimotor context is defined as a behavior combined with a sensory modality, e.g., *drop-audio*. We will use the notation *behavior-modality* to denote a context and the letter $\mathcal{C}$ to denote the set of all contexts. Table 6.1 shows all combinations of behaviors and modalities that the robot used. The robot used all



(a) Color: green, red, and blue.

(b) Contents: glass, rice, beans, and screws.

(c) Weight: light, medium, and heavy.

Figure 6.2: The properties by which the objects varied. Each object is a jar that is one of three colors, filled with one of four different types of contents, and weighing one of three different weights, for a total of 36 objects (see Figure 6.3).

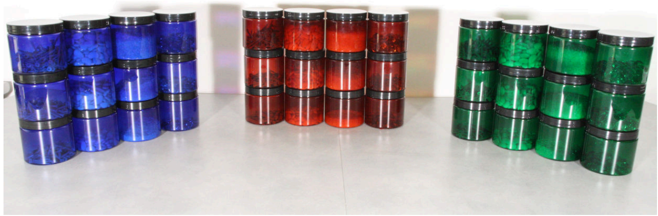
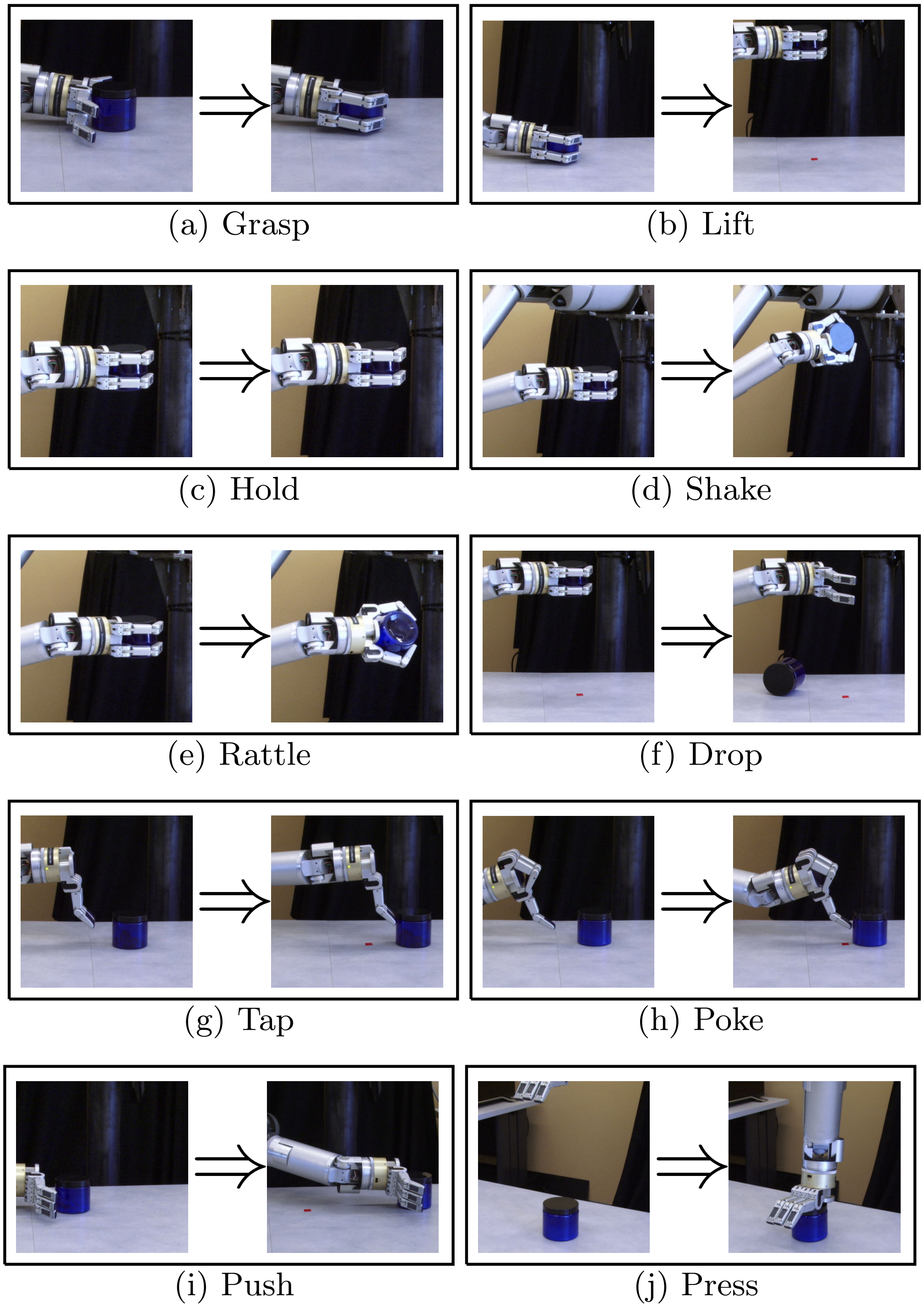


Figure 6.3: The 36 objects used in the experiments described in this chapter, grouped by color. Within each group, all objects of the same weight are in the same row and all objects with the same type of contents are in the same column.



(a) Grasp

(b) Lift

(c) Hold

(d) Shake

(e) Rattle

(f) Drop

(g) Tap

(h) Poke

(i) Push

(j) Press

Figure 6.4: Before and after images for the ten exploratory behaviors that the robot performed on all objects. From left to right and top to bottom: *grasp*, *lift*, *hold*, *shake*, *rattle*, *drop*, *tap*, *poke*, *push*, and *press*. The object was placed back in the initial position by the experimenter after some of the behaviors (e.g., drop).

Table 6.1: The set of sensorimotor contexts used by the robot. The X's denote modality-behavior combinations that the robot used to solve matrix completion tasks.

Behavior	**Modality**		
	proprioception	*audio*	*color*
look			**X**
grasp	**X**	**X**	
lift	**X**	**X**	
hold	**X**	**X**	
shake	**X**	**X**	
rattle	**X**	**X**	
drop	**X**	**X**	
tap	**X**	**X**	
poke	**X**	**X**	
push	**X**	**X**	
press	**X**	**X**	

behaviors except *look* in combination with the modalities *audio* and *proprioception*. In addition to this, the robot also used the context *color-look*. This resulted in a total of $|\mathcal{C}| = 10\times2+1 = 21$ sensorimotor contexts.

For each object O_i and each context $c \in \mathcal{C}$, a set of feature vectors $\mathcal{X}_i^c$, was computed as described below in section 6.3. Each $\mathbf{x} \in \mathcal{X}_i^c$ is a feature vector computed from one interaction in context c . Because each behavior was performed 10 times on each object, there were 10 feature vectors in each $\mathcal{X}_i^c$, i.e. $|\mathcal{X}_i^c| = 10$.

In the previous two chapters (Chapters 4 and 5) we used only 2 sensory modalities (*audio* and *proprioception*) and 10 behaviors for a total of 20 contexts. In those chapters, vision was not required to solve the tasks. In this chapter, however, color is an important property of the objects, so the robot was required to use vision to solve the tasks. Thus, we added the *color-look* context for a total of 21 contexts (the same 20 from the previous chapters plus *color-look*).

6.2.5 Data Collection

The robot interacted with the objects by performing each of the behaviors on each object ten times. At the start of this process the experimenter placed the first object at a specified location on the table in front of the robot, and then the robot performed one of its behaviors on the object. The experimenter then placed the second object on the table in the same spot, and the robot performed the same behavior on it. This was repeated for all objects. The experimenter then placed the first object in the same spot on the table and the robot performed the next behavior on it, repeating this again for all objects. This was done until the robot had performed each behavior once on each object. This entire process was then repeated nine more times (for a total of ten repetitions) such that the robot had performed each behavior ten times on each object. There were 36 objects, 10 behaviors, and 10 repetitions, resulting in a total of $36 \times 10 \times 10 = 3600$ interactions with the objects.

6.3 Feature Extraction

6.3.1 Proprioceptive Feature Extraction

During each interaction, the robot recorded the joint torques applied to its right arm. The robot sampled from all 7 joints at 500 Hz. This resulted in $7 \times m$ real numbers, where m is the number of temporal samples during each interaction. In other words, each interaction resulted in a matrix, where each column contains the joint torque readings at one point in time and each row contains the torque values applied to one joint over the course of the interaction. This matrix was too high-dimensional to be effective for the tasks in this chapter, so features were extracted from it by binning the real values for each joint into 10 temporal bins. That is, the first $\frac{m}{10}$ columns were summed together into one column, then the second $\frac{m}{10}$ were summed together, and so on. This resulted in a feature vector $\mathbf{x} \in \mathbb{R}^{7\times 10}$. Figure 6.5 illustrates this process.

6.3.2 Auditory Feature Extraction

During each interaction, auditory data was recorded by the microphones in the robot's head in the form of a wave file. Each wave file was then converted into a spectrogram using the log-normalized Discrete Fourier Transform (DFT) with $2^5 + 1 = 33$ frequency bins. The SPHINX4 natural language processing library was used to compute the DFT for each audio

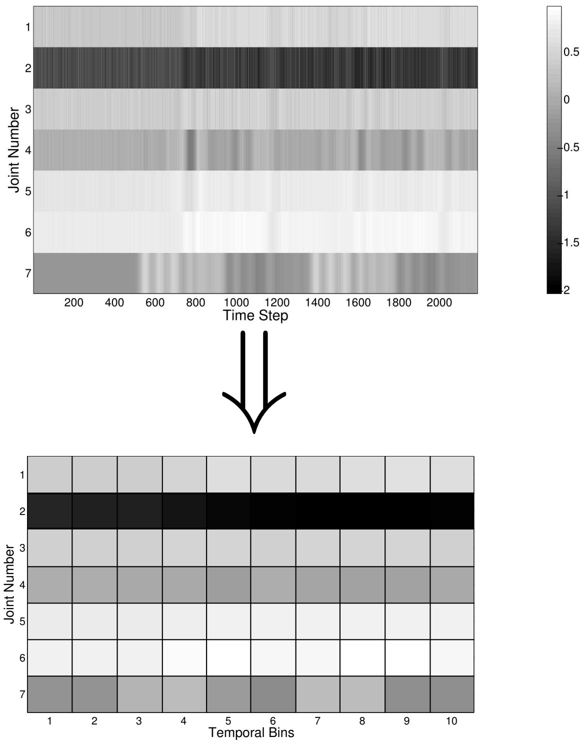


Figure 6.5: An example sensory record of proprioceptive values. The top image depicts the raw joint torques recorded from the robot's arm during the interaction where light values denote positive torque values and dark values indicate negative torque values. The bottom image depicts the features extraced from that by binning the values for each of the 7 joints into 10 temporal bins.

file (Lee et al., 1990). The spectrogram for each audio file was computed by computing the DFT over the length of the interaction. This resulted in a 33 × m dimensional matrix, where m is the number of time samples. Like in the proprioception case, this matrix was too high-dimensional to be useful. To lower the dimensionality, a 10 × 10 spectro-temporal histogram was computed for each of the audio spectrograms. That is, each spectrogram was divided up into $10 \times 10 = 100$ bins and then all the values in each bin were summed. This resulted in a feature vector $\mathbf{x} \in \mathbb{R}^{10\times 10}$. An example of this process is shown in Figure 6.6. In addition to the temporal binning done in the previous method, this method also performed frequency binning.

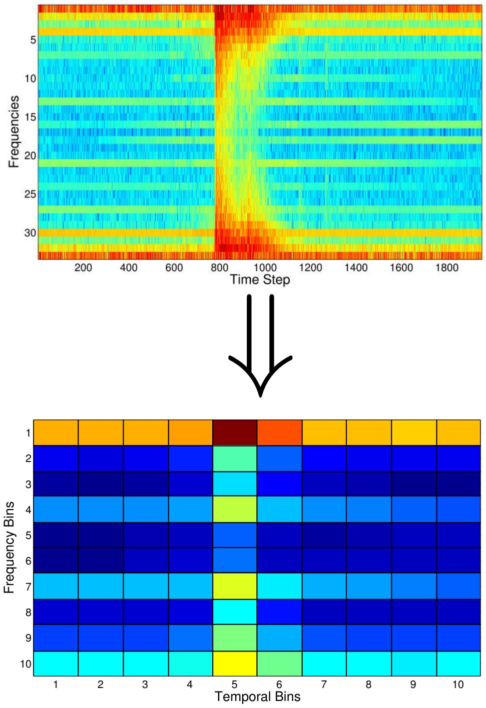


Figure 6.6: An example sensory record of auditory values. The top image depicts the audio spectrogram, which was computed using a series of DFTs on the raw wave that was recorded during the interaction (red denotes higher activation, green and blue denote lower activation). The bottom image depicts the features extracted from the spectrogram by binning the values into 10 temporal bins and 10 frequency bins.

6.3.3 Visual Feature Extraction

Visual features were computed based on the color information from the robot's Kinect camera. For simplicity, color features were only extracted during the behavior *look* (based on the methodology described by Sinapov et al. (2013)). During each *look* behavior, the robot recorded a short series of images of the object sitting on the table in front of it. The object was always placed in approximately the same spot on the table, so it was segmented out of each recorded image using a pre-set region of interest. For each image, the robot then divided this region into $r \times r$ bins (where $r = 8$) and averaged the HSV values in each bin, resulting in a vector $\mathbf{x}_i \in \mathbb{R}^{r \times r \times 3}$. This process is shown in Figure 6.7. For each image in a series of images from one *look* behavior, the robot simply averaged the vectors together as follows:

$$\mathbf{x} = \frac{1}{k}\sum_{i=1}^{k}\mathbf{x}_i$$

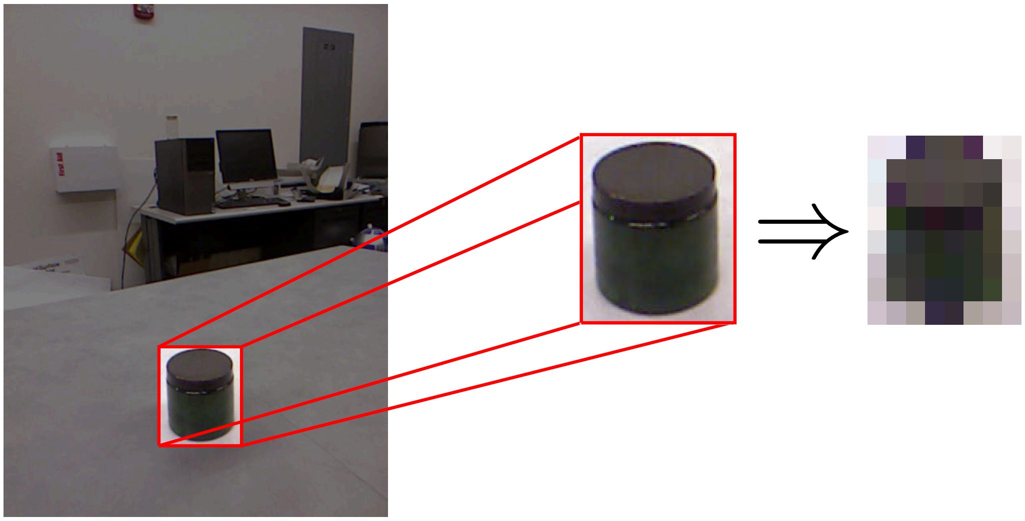


Figure 6.7: An example image recorded during the look behavior. The left image shows the raw RGB data that the robot's camera recorded. The middle image is the segment of the robot's field of view where the object was always placed on the table. The right image is the 8×8 grid that this segment was binned into, where each location in the grid is the average of the pixels that fall into that cell.

where k is the number of images captured during the *look* behavior. This resulted in a feature vector $\mathbf{x} \in \mathbb{R}^{r \times r \times 3}$ for each *look* behavior.

6.4 Experimental Methodology

6.4.1 Problem Formulation

Let $\mathcal{M}$ denote a matrix of objects with n rows and n columns[1] where $\mathcal{M}_{ij}$ is the object in the i-th row and the j-th column. Let R_i be the i-th row of $\mathcal{M}$ and C_j be the j-th column of $\mathcal{M}$. Let $\mathcal{P}^r$ and $\mathcal{P}^c$ be sets of patterns defined over the rows and columns, respectively. For all $p \in \mathcal{P}^r$, let $f_p^{\Rightarrow}$ denote a binary function that takes as input a row of $\mathcal{M}$ and evaluates to true if pattern p is present in that row and false otherwise. The binary function $f_p^{\Downarrow}$ is similarly defined for all $p \in \mathcal{P}^c$.

Let $present^{\Rightarrow}$ be a binary function defined over matrices and patterns. Given a pattern

[1] In these experiments we used only square matrices, but it is easy to extend this methodology to non-square matrices.

$p \in \mathcal{P}^r$ and a matrix $\mathcal{M}$, it is evaluated as follows:

$$present^{\Rightarrow}(p, \mathcal{M}) = \begin{cases} true, & \text{if } \forall_{(\mathcal{R}_i \in \mathcal{M} - \{\mathcal{R}_n\})} \ f_p^{\Rightarrow}(\mathcal{R}_i) = true \\ false, & \text{otherwise.} \end{cases}$$

In other words, this function evaluates to true if and only if p is present in all but the last row of $\mathcal{M}$. The function $present^{\Downarrow}$ be similarly defined for all $p \in \mathcal{P}^c$.

A matrix $\mathcal{M}$ is considered valid with respect to $\mathcal{P}^r$ and $\mathcal{P}^c$ if and only if it satisfies the following conditions:

$$\forall_{p \in \mathcal{P}^r} \ present^{\Rightarrow}(p, \mathcal{M}) \rightarrow f_p^{\Rightarrow}(\mathcal{R}_n) = true$$

$$\forall_{p \in \mathcal{P}^c} \ present^{\Downarrow}(p, \mathcal{M}) \rightarrow f_p^{\Downarrow}(\mathcal{C}_n) = true$$

The first condition enforces that if the pattern $p \in \mathcal{P}^r$ exists in the first $n-1$ rows of the matrix, then it must also exist in the last row. The second condition enforces the same constraint, but on the columns. The idea behind this is that the patterns detected in the first $n-1$ rows and columns can be used to determine the candidate object that best fits in the last spot in the matrix.

A matrix completion task is defined as the ordered pair $(\mathcal{M}, \mathcal{G})$ where $\mathcal{M}$ is a matrix that is missing the object $\mathcal{M}_{n,n}$ (i.e., the object in the lower-right corner), and $\mathcal{G}$ denotes a set of candidate objects such that exactly one object may be placed in the empty space and cause $\mathcal{M}$ to be a valid matrix as defined above. In other words, the task is to select the object from $\mathcal{G}$ that creates a valid matrix with respect to the patterns in $\mathcal{P}^r$ and $\mathcal{P}^c$.

6.4.2 Task Generation

To generate a set of matrix completion tasks to test the robot on, we first had to generate two sets of patterns $\mathcal{P}^r$ and $\mathcal{P}^c$ using the generation rules described in Section 2.4.4 in Chapter 2. Once again, those rules are: *constant*, *increment*, *decrement*, and *permutation*. Let a pattern be defined as an instantiated rule, i.e., a rule and a property that the rule applies to. For example, the rule *constant* applied to the property *color* would be denoted by *constant:color* and would mean that the color of the entries in the row or column is constant. In general, we will use the notation *rule:property* to denote patterns in the matrix.

The objects in these experiments vary by three properties: *color*, *contents*, and *weight*. To determine the patterns to use, we applied the rule *constant* to all three properties, the rules *increment*/*decrement* to the ordered properties (*weight*), and the rule *permutation* to the unordered properties (*color* and *contents*). This resulted in seven patterns: *constant:contents*, *constant:color*, *constant:weight*, *permutation:contents*, *permutation:color*, *increment:weight*, and *decrement:weight*. The same set of patterns were used for both the rows and the columns.

The matrix completion tasks were generated as follows. Two patterns, p^r and p^c, were randomly selected from the set of patterns for rows, P^r, and columns, P^c, respectively. A matrix $\mathcal{M}$ was randomly selected from the set of all valid matrices such that p^r was present in all the rows of $\mathcal{M}$ and p^c was present in all the columns. Seven objects were then randomly selected from the set of objects not in $\mathcal{M}$ such that none of them could replace the last object in $\mathcal{M}$ and create a valid matrix. These objects comprised the set $\mathcal{G}$. The object in the lower-right corner of $\mathcal{M}$ was then removed from $\mathcal{M}$ and placed in $\mathcal{G}$. The resulting matrix completion task was the ordered pair $(\mathcal{M}, \mathcal{G})$.

It is worthwhile to mention that, because the number of valid matrices is exponentially large, the above algorithm is intractable. Therefore, we used a slightly different algorithm that is functionally equivalent to that algorithm. We divided the set of patterns into groups, one for each property. For example, all the patterns over *color* in one group (*constant:color*, *permutation:color*), all the patterns over *contents* in another group (*constant:contents*, *permutation:contents*), etc. For each group, we iterated over all possible permutations of the values of length $3 \times 3 = 9$ of the associated property, keeping only the permutations that correspond to valid matrices with respect to the group of patterns (that is, matrices that are valid if only that one property is considered). While the number of possible permutations for even a single property is also exponentially large (e.g., because there are 3 values for *color*, there are $3^9 = 19,683$ possible permutations), it is tractable when the size of the matrix and the number of values for the properties are relatively small (which in this case, they are). We were then able to randomly select one valid permutation from each of these sets and combine each of them together. Doing so resulted in a complete matrix. That is, since there was one permutation for each property, then the property values for each object in the matrix were specified by combing each of the permutations together. This uniquely specified the object that belonged in each location in the matrix. This allowed us to randomly sample from the set of all possible valid matrices.Given this, we could then generate the tasks as described in the previous paragraph. The next section describes how the robot selects the best object to complete each task.

6.4.3 Selecting the Best Candidate to Complete a Matrix

Given a matrix reasoning task $(\mathcal{M}, \mathcal{G})$, the robot must select the best candidate object from the set $\mathcal{G}$ to complete the matrix $\mathcal{M}$. In order to do this, the robot first generates a set of distance functions $\mathcal{D}$ (described below in section 6.4.5) such that the value of $D(O_i, O_j) \in [0, 1]$ is the distance between object O_i and object O_j as measured by $D \in \mathcal{D}$.

Next, the robot selects the best candidate object from $\mathcal{G}$ by finding the object $O_k \in \mathcal{G}$ that minimizes the following objective function:

$$q(O_k, \mathcal{M}, \mathcal{D}) = \sum_{D \in \mathcal{D}} A_D \left[\sum_{j=1}^{n-1} \Big(D(\mathcal{M}_{n,j}, O_k) - \mathbf{E}[D(\mathcal{M}_{n,j}, \widehat{\mathcal{M}}_{n,n})] \Big)^2 \right], \tag{6.1}$$

where $\mathcal{M}$ is an $n \times n$ matrix that is missing its lower-right element, A_D is the consistency of D across the rows of $\mathcal{M}$, and $\mathbf{E}[D(\mathcal{M}_{n,j}, \widehat{\mathcal{M}}_{n,n})]$ is the expected distance between $\mathcal{M}_{n,j}$ and $\widehat{\mathcal{M}}_{n,n}$ with respect to D. In this formula $\widehat{\mathcal{M}}_{n,n}$ represents the robot's estimation of the missing object, so the expected distance is computed, rather than the actual distance, because the object is missing. The intuition behind this function is that the robot computes the difference between the object that *should* be in the missing space and O_k. It does this by computing the squared difference between what it expects D to evaluate to and what it evaluates to when placing O_k in the missing spot, for all $D \in \mathcal{D}$.

In equation (6.1), A_D denotes the consistency for the distance function D. A consistent distance function is one in which objects in the same relative positions, but in different rows, vary in the same manner. For example, the first and second objects in each row are always the same distance apart for D, regardless of the row. It is assumed that the more consistent a distance function is (values closer to 1 for A_D), the more useful that function is for solving the task. Conversely, the more inconsistent a distance function is (values closer to 0 for A_D), the less useful that function is for solving the task. Thus, A_D acts as a task-specific weight, allowing the robot to isolate the distance functions that vary in the most consistent way in the matrix. It should be noted, though, that the objective function q, as defined in equation (6.1), only evaluates patterns across the rows and not down the columns. In section 6.4.4 we will extend this function to also evaluate patterns down the columns of $\mathcal{M}$.

The expectation $\mathbf{E}[D(\mathcal{M}_{n,j}, \widehat{\mathcal{M}}_{n,n})]$ is computed as follows:

$$\mathbf{E}[D(\mathcal{M}_{n,j}, \widehat{\mathcal{M}}_{n,n})] = \frac{1}{n-1}\left[\sum_{i=1}^{n-1} D(\mathcal{M}_{i,j}, \mathcal{M}_{i,n})\right]. \tag{6.2}$$

Intuitively, this expectation is simply the average distance computed using D between pairs of objects in the same relative positions in every row except the last one.

The consistency, A_D, of a distance function D with respect to a matrix $\mathcal{M}$ is measured as:

$$A_D = \prod_{a=1}^{n-1} \prod_{b=a+1}^{n-1} A_D^{a,b}. \tag{6.3}$$

In equation (6.3) $A_D^{a,b}$ is the consistency between rows a and b, which is defined as:

$$A_D^{a,b} = \prod_{i=1}^{n} \prod_{j=i+1}^{n} h(\left|D(\mathcal{M}_{a,i}, \mathcal{M}_{a,j}) - D(\mathcal{M}_{b,i}, \mathcal{M}_{b,j})\right|), \tag{6.4}$$

where h is the consistency function. Thus, A_D measures how often D agrees with itself for two pairs of objects in the same relative positions but in different rows. The consistency function h is defined as[2]:

$$h(x) = 1 - \log_2(x + 1).$$

To summarize, in order to solve the matrix completion task, the robot selects the object in $\mathcal{G}$ that minimizes the squared difference between that object and the expectation based on the consistency of the distance functions in $\mathcal{D}$ with respect to the matrix $\mathcal{M}$.

The asymptotic running time to compute this objective function for a given matrix $\mathcal{M}$ and a given candidate object O_k is: $O\left(|\mathcal{D}| \times n^4\right)$ where $|\mathcal{D}|$ is the size of the set of distance functions and n is the size of the matrix (that is, a square matrix with n rows and n columns). The $|\mathcal{D}|$ term is due to the first summation in equation (6.1) over the set of distance functions. The term n^4 is due largely to the computation of A_D. Computing each sub-term $A_D^{a,b}$ takes

[2] The consistency function was empirically determined based on the condition that its output should be maximized when given a minimal disagreement value (i.e., $x = 0$) and its output should be minimized when given a maximal disagreement value (i.e., $x = 1$).

$O(n^2)$ time and there are $O(n^2)$ of them to compute, thus it takes $O(n^2) \times O(n^2) = O(n^4)$ time to compute A_D. Comparatively, it takes $O(n^2)$ to compute the inner summation in equation (6.1). Thus, computing A_D dominates the running time of the function, but with relatively small n (in these experiments $n = 3$), the running time is still fairly short.

6.4.4 Extending the Methodology to Columns

The methodology described so far only considers relationships between objects in the same row. In most common matrix completion tests, however, the relationships between the objects down the columns are just as important for solving the tasks. One way to alter the methodology to work with column-wise relationships is to simply transpose the matrix and use the same objective function (i.e., evaluate $q(O_k, \mathcal{M}^T, \mathcal{D})$). Since the transpose operator flips the columns and the rows, the modified objective function evaluates the relationships between the objects down the columns.

It is not enough, though, to just evaluate the relationships down the columns or across the rows of a matrix independently. Most matrix reasoning tests require that the test taker be able to combine these two together in order to pick the correct answer. In order to do this, we define the super objective function Q to be equal to:

$$Q(O_k, \mathcal{M}, \mathcal{D}) = q(O_k, \mathcal{M}, \mathcal{D}) + q(O_k, \mathcal{M}^T, \mathcal{D}).$$

In other words, Q simply sums the values of the objective function q for a given object O_k and a set of distance functions $\mathcal{D}$ across the rows and down the columns. In this way, the best candidate object is defined as the one that best approximates the relationships between the objects in the matrix down the rows *and* across the columns.

6.4.5 Measuring Object Similarity

Four different methods for generating the set of distance functions were evaluated. The first was simply the Euclidean distance between the features for each object in each sensorimotor context. The second used spectral clustering to group the objects into labeled categories and then measured the distance between the objects based on their category memberships. The third was an extension of the first; it added supervision to the context distances in an attempt to improve the performance. The fourth method was similar to the third; it added supervision to the second method in an attempt to improve performance.

6.4.5.1 Context Distance Measurements

One distance function D_c was computed for each sensorimotor context $c \in \mathcal{C}$. Given two objects, O_i and O_j, the output of D_c is defined as:

$$D_c(O_i, O_j) = \mathbf{E}\Big[\|\mathbf{x}_a - \mathbf{x}_b\| \Big| \mathbf{x}_a \in \mathcal{X}_i^c, \mathbf{x}_b \in \mathcal{X}_j^c\Big],$$

where $\|\mathbf{x}_a - \mathbf{x}_b\|$ is the L2-norm distance between $\mathbf{x}_a$ and $\mathbf{x}_b$ and $\mathcal{X}_i^c$ and $\mathcal{X}_j^c$ are two sets of feature vectors for context c for O_i and O_j respectively (recall that the robot repeated the same behavior multiple times on each object). This expectation is estimated by:

$$D_c(O_i, O_j) = \frac{1}{|\mathcal{X}_i^c| \times |\mathcal{X}_j^c|} \sum_{\mathbf{x}_a \in \mathcal{X}_i^c} \sum_{\mathbf{x}_b \in \mathcal{X}_j^c} \|\mathbf{x}_a - \mathbf{x}_b\|.$$

To mitigate the effect of outliers, the output of each distance function D_c was normalized to be between 0 and 1 using the logistics function, $\frac{1}{1+e^{-x}}$, with the middle two quartiles of the comparisons falling in the range $[0.1, 0.9]$. The resulting set $\mathcal{D}$ contained exactly one distance function D_c for each context $c \in \mathcal{C}$.

This method for computing context distance measurements is identical to the one used in Chapter 5. For more details see Section 5.3.2.

6.4.5.2 Category Distance Measurements

As before, one distance function D_c was computed for each sensorimotor context $c \in \mathcal{C}$. For each context, the spectral clustering algorithm (Ng et al., 2002) was used to cluster the objects[3] into a set of categories. Given a set of categories, the output of D_c was computed as

$$D_c(O_i, O_j) = 1 - I(label_c(O_i) \equiv label_c(O_j)),$$

[3] The spectral clustering algorithm requires a distance function between the datapoints in order to cluster them. Since the robot created a separate clustering for each sensorimotor context, it computed the distance between each pair of objects in each context in the same way as described in Section 6.4.5.1.

where $label_c(O_i)$ is the category label for O_i in context c and I is the indicator function, which is 1 if its argument is true and 0 otherwise. Intuitively, the output of the distance function is 0 if the two objects belong to the same category in a specific sensorimotor context and 1 if they don't.

6.4.5.3 Context Distance Measurements with Supervision

This method builds on the context distances method. Given the set of distance functions $\mathcal{D}$ computed from that method and a set $\mathcal{L} = \{(\mathcal{M}_1, \mathcal{G}_1), ..., (\mathcal{M}_n, \mathcal{G}_n)\}$ of training matrix completion tasks, for which the correct answers are known, the robot attempted to find the set $\mathcal{D}' \subseteq \mathcal{D}$ that maximized performance on the training set. That is, it attempted to prune the set $\mathcal{D}$ down to only the most useful functions. It did this by iteratively removing the worst distance function from the set, starting with all distance functions and ending when all but one function have been removed. It then returns the subset of distance functions that has the best performance on the training set. This algorithm is shown in more detail in Figure 6.8.

This algorithm relies on the assumption that some of the computed distance functions are not useful for solving matrix completion tasks. Unlike the variable A_D in the objective function (equation (6.1)), which weights each distance function based on its consistency for an *individual* task, this algorithm attempts to find distance functions that are not useful *across all tasks* for solving matrix completion tasks and removes them from consideration. This is different from the previous chapters (Chapters 4 and 5), where the robot weighted each context (analogous to the distance functions used here) based on the individual performance of each context on a training set of tasks. In both of those chapters we found that certain individual contexts performed near perfectly on certain types of tasks on their own because the objects in them largely varied by a single property. In this chapter, the objects vary by multiple properties, and as a result we empirically determined that similar weighting schemes would not work because no individual distance function performed even moderately well by itself on the training set of tasks. Thus, we developed this algorithm as a solution to that problem.

```
function PRUNE($\mathcal{D}$, $\mathcal{L}$)
    $\mathcal{D}'[\,] \leftarrow emptyArray$
    $count \leftarrow 0$
    $\mathcal{D}'[count] \leftarrow \mathcal{D}$
    while $|\mathcal{D}'[count]| > 1$ do
        $bestPerformance \leftarrow 0$
        $bestSet \leftarrow null$
        for all $D \in \mathcal{D}'[count]$ do
            $set \leftarrow \mathcal{D}'[count] - \{D\}$
            $p \leftarrow evaluatePerformance(set, \mathcal{L})$
            if $p > bestPerformance$ then
                $bestPerformance \leftarrow p$
                $bestSet \leftarrow set$
            end if
        end for
        $\mathcal{D}'[count + 1] \leftarrow bestSet$
        $count \leftarrow count + 1$
    end while
    $bestPerformance \leftarrow 0$
    $bestSet \leftarrow null$
    for $i \leftarrow 0$ to $length(\mathcal{D}') - 1$ do
        $p \leftarrow evaluatePerformance(\mathcal{D}'[i], \mathcal{L})$
        if $p > bestPerformance$ then
            $bestPerformance \leftarrow p$
            $bestSet \leftarrow \mathcal{D}'[i]$
        end if
    end for
    return $bestSet$
end function
```

Figure 6.8: The algorithm that prunes the set of distance functions. It takes as input an initial set of distance functions $\mathcal{D}$ and a set of training tasks $\mathcal{L}$ for which the correct answers are known. The method *evaluatePerformance* returns the accuracy of the given set of distance functions on the given training tasks.

6.4.5.4 Category Distance Measurements with Supervision

This method is similar to the previous method except that it builds on the category distances method rather than the context distances method. The robot first uses the spectral clustering algorithm to cluster the set of objects into a set of categories, one set for each context. Next, given the set of distance functions $\mathcal{D}$ computed from these sets of categories (as described in Section 6.4.5.2) and a set of matrix completion tasks for training $\mathcal{L} = \{(\mathcal{M}_1, \mathcal{G}_1), ..., (\mathcal{M}_n, \mathcal{G}_n)\}$, for which the correct answers are known, the robot again attempted to prune the set of distance functions down to only the most useful. Similar to the last method, it did this by iteratively removing the worst performing distance function from the set until it found the best subset of functions. It used the same algorithm as before, which is described in Figure 6.8.

6.4.6 Evaluation

The robot was evaluated on the set of objects described in Section 6.2.2, which vary by *color*, *contents*, and *weight*. The values for *color* are *red*, *blue*, and *green*. The values for *contents* are *glass*, *screws*, *beans*, and *rice*. The values for *weight* are *light*, *medium*, and *heavy*. It should be noted that the robot was never given these values during the evaluation.

We randomly generated 500 matrix completion tasks using the methodology described in Section 6.4.2. The robot then generated each of the four sets of distance functions described in Section 6.4.5. Each set was evaluated independently. Because some distance functions required training, we performed 10-fold cross-validation across the matrix completion tasks. That is, we split the 500 tasks into 10 equally sized groups, trained the robot on 9 of the 10 groups, and tested it on the remaining one. This process was repeated for each group. It is worthwhile to note that the robot was never given *a priori* knowledge of the patterns used to generate the matrix completion tasks. Rather, the only supervision it was given was in the form of example matrix reasoning tasks.

Performance is reported as accuracy or kappa. The accuracy is computed as

$$\%Accuracy = \frac{\#correct\ answers}{\#total\ tasks} \times 100.$$

We also wanted to know how the robot performs when varying the number of candidate objects to choose from. Since chance accuracy depends on the number of candidate objects in the task, the kappa score was computed to compensate for varying degrees of chance. Cohen's kappa statistic (Cohen, 1960) was computed as follows:

$$kappa = \frac{P(a) - P(e)}{1 - P(e)},$$

where $P(a)$ is the performance of the robot's model and $P(e)$ is chance accuracy. This allows the direct comparison of results where chance accuracy may differ.

The evaluation was performed off-line after the robot interacted with all 36 objects..

6.5 Results

6.5.1 Performance on a Single Task

Figure 6.9 illustrates one of the 500 matrix completion tasks that the robot solved. The objective function values for each of the 8 candidate objects are shown for both the context and category distance functions with and without supervision. The matrix in the figure exhibits the patterns *constant:color* and *decrement:weight* across its rows and *permutation:color* and *constant:weight* down its columns. Given this, it can be deduced that the missing object must be *light* and *red*. The only candidate object that has both of these property values is (g), which is the correct answer. In this case, the property *contents* was irrelevant to the task.

The objective function values were computed using all 21 contexts. The context distance method ranked three candidate objects ((d), (e), and (a)) higher than the correct answer (g). The category distance method performed about the same, ranking (c), (d), and (e) higher than the correct answer. Thus, both of the unsupervised methods failed to pick the correct answer.

The two supervised[4] methods, however, performed better. The supervised context distances method ranked the correct object, (g), in second place after (d). The supervised category distance method picked the correct answer, ranking (g) significantly higher than any other candidate. It selected the distance functions computed from the contexts *audio-lift*, *audio-hold*, *audio-rattle*, *audio-push*, *proprioception-rattle*, and *color-look*. This indicates, as expected, that

[4] For the example task, both of the supervised methods were trained on 450 randomly selected tasks from the set of 500 generated for this chapter. In other words, they were trained on 9 of the 10 folds. The example task shown in Figure 6.9 was not included in that training set.

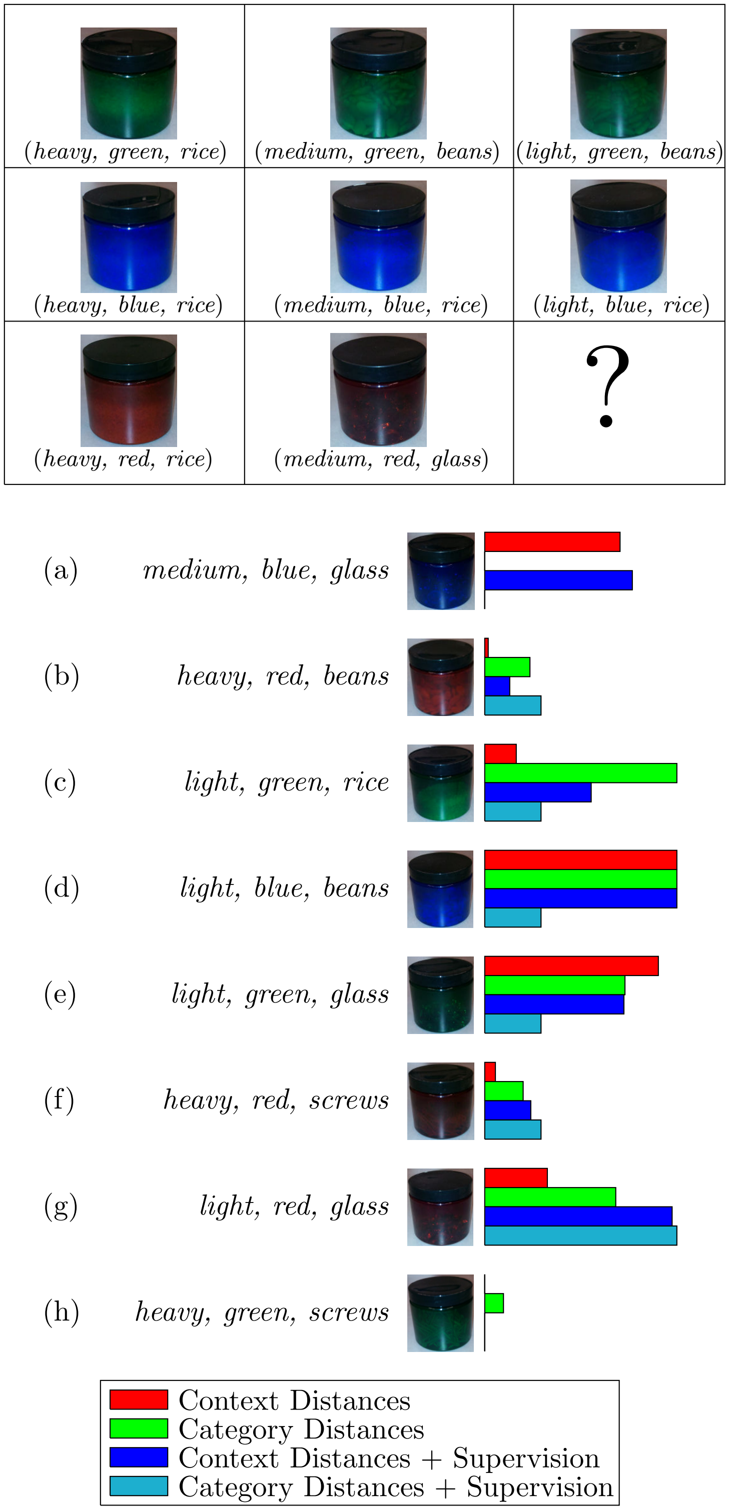


Figure 6.9: An example matrix reasoning task solved by the robot as part of this experiment. The words below each object in the matrix represent the values for each of the three properties for that object. The bars next to each candidate object represent the normalized objective function values for each of the four distance methods. The correct answer is (g).

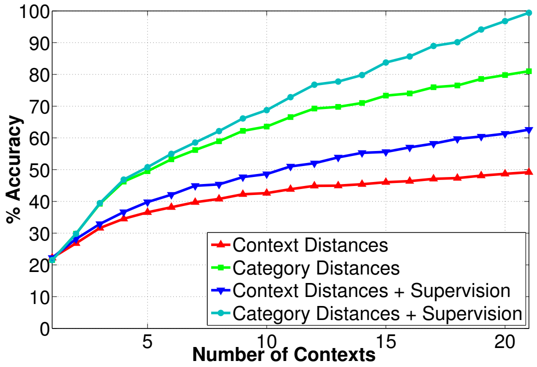


Figure 6.10: Accuracy versus number of contexts used to solve the tasks. As expected, the accuracy improves as the robot is allowed to use information from more sensorimotor contexts. Each line represents a different distance function method. The two category distance methods perform better than the two context distance methods. As expected, the category method with supervision performs the best.

not all contexts are useful for solving this type of matrix completion task. This suggests that methods that use supervision to prune the contexts to the most useful ones could perform better. The next section expands on this by looking at performance over all 500 tasks.

6.5.2 Performance Across All Tasks

Figure 6.10 compares the accuracy of all four methods of measuring distances. Just like we found in the last two chapters, the robot's accuracy on matrix completion tasks improves when it is given access to more information in the form of sensorimotor contexts. It is interesting to note that both of the category distances methods (with and without supervision) perform the best. This suggests that features derived from category labels are more useful for this kind of task. Additionally, for both category and context distances, the method with supervision always outperforms its unsupervised counterpart. The best performing method was the category method with supervision. When given access to all 21 contexts, it was able to achieve 99.4% accuracy on the testing set of matrices. That is, using all available information and the best performing method, the robot was able to determine the correct answer to all but 3 of the 500 problems that were presented to it.

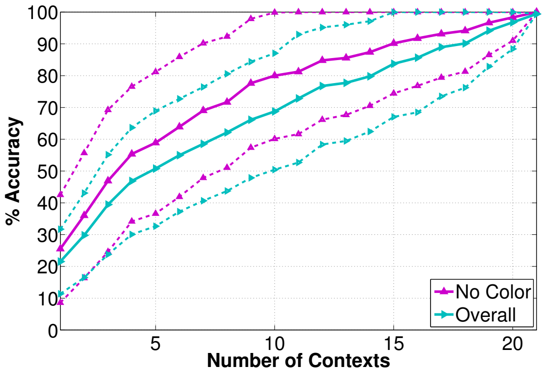
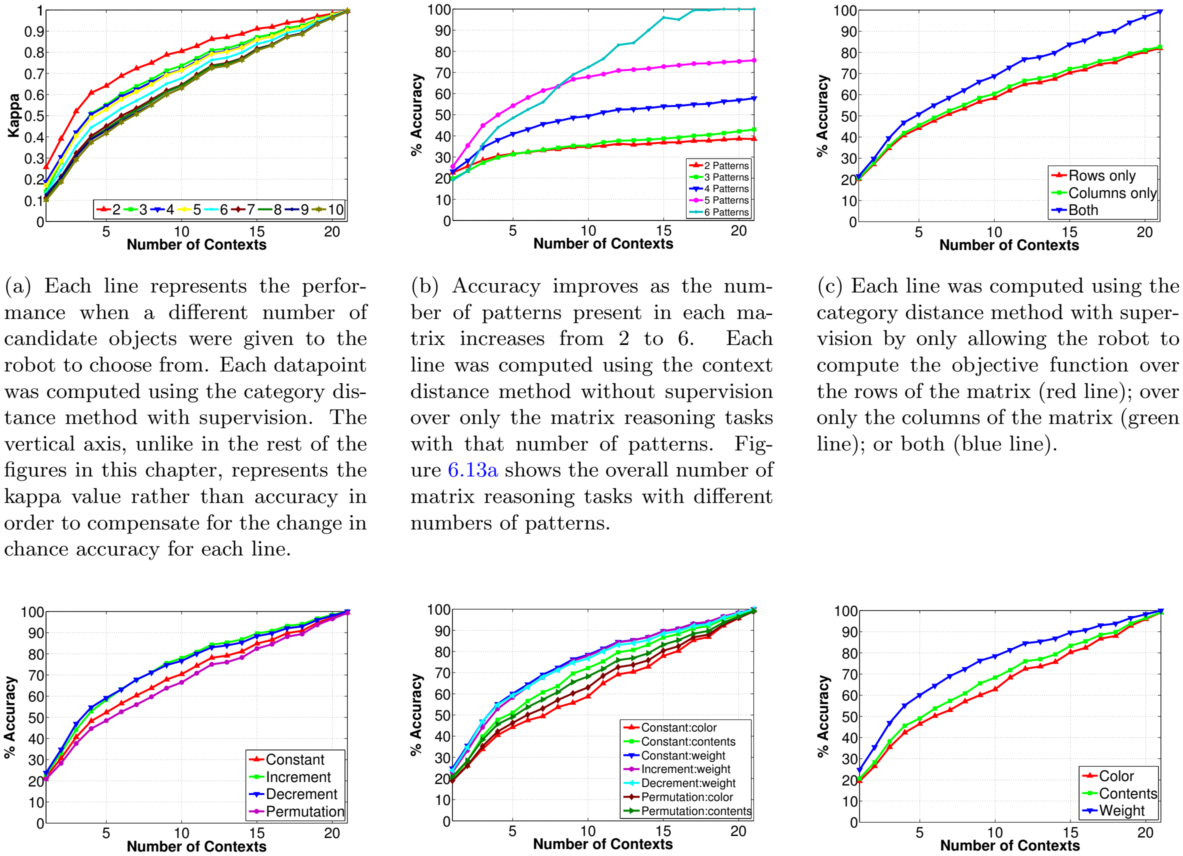


Figure 6.11: Accuracy versus number of contents for the subset of tasks that don't require color information and for all tasks. The line labeled "Overall" is the same as the line labeled "Category Distances + Supervision" in Figure 6.10. The other line was computed in the exact same way as the overall line with the exception that the 500 tasks were reduced to just the 173 that did not require perception of color to solve. The standard deviation for each data point is also plotted using dashed lines.

As described in Section 6.2.4, there was only one context, *color-look*, that had access to visual data collected from the robot's camera. Because color is so important to solving the tasks, we wanted to know how this affected the robot's performance. Figure 6.11 shows the robot's performance on only the matrix completion tasks that did not require the perception of color to solve (173 out of 500) as compared to the robot's performance on all 500 tasks. On average the robot performs better when the task does not involve color, especially in the middle part of the graph (5 to 15 contexts). It is also worth mentioning that the upper limit of the standard deviation converges to 100% accuracy sooner for tasks not involving color than for all tasks. Just as we expected, because there are no redundant contexts in which *color* can be perceived (as opposed to *weight* and *contents*), the robot has a harder time identifying color as a relevant property, and thus tasks that require it are harder to solve.

6.5.3 Performance Compared to Difficulty of the Task

Figure 6.12 shows six figures that compare the robot's performance for different types of task difficulty. Figure 6.12a shows the robot's performance as a function of the number of



(a) Each line represents the performance when a different number of candidate objects were given to the robot to choose from. Each datapoint was computed using the category distance method with supervision. The vertical axis, unlike in the rest of the figures in this chapter, represents the kappa value rather than accuracy in order to compensate for the change in chance accuracy for each line.

(b) Accuracy improves as the number of patterns present in each matrix increases from 2 to 6. Each line was computed using the context distance method without supervision over only the matrix reasoning tasks with that number of patterns. Figure 6.13a shows the overall number of matrix reasoning tasks with different numbers of patterns.

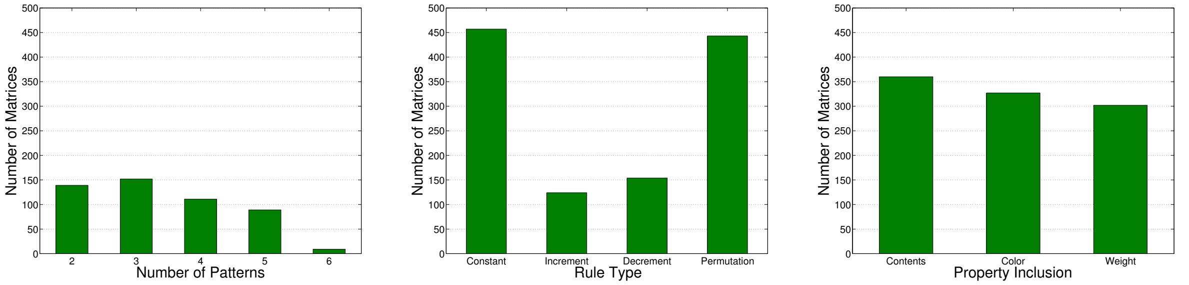
(c) Each line was computed using the category distance method with supervision by only allowing the robot to compute the objective function over the rows of the matrix (red line); over only the columns of the matrix (green line); or both (blue line).

(d) Each line was computed using the category distance method with supervision over only the tasks that had at least one instance of the corresponding rule (e.g., the red line was computed using only tasks that contained the rule *constant*). See also Figure 6.13b

(e) Each line was computed using the category distance method with supervision over only the tasks that had the corresponding pattern present (e.g., the increment:weight line was computed only over tasks that contained the pattern *increment:weight*).

(f) Each line was computed using the category distance method with supervision over only the tasks that had at least one pattern over the corresponding property (e.g., the color line was computed only over tasks that contained at least one pattern over *color*). See also Figure 6.13c.

Figure 6.12: Six figures that compare the performance as a function of the difficulty of the matrix completion tasks.



(a) The number of tasks that have 2 to 6 patterns.

(b) The number of tasks that have at least one instance of each of the four rules.

(c) The number of tasks that include at least one pattern over each object property.

Figure 6.13: Three figures that show the number of matrix completion tasks for different task difficulty types.

candidate objects that it can choose from to complete the matrix. It is interesting to note that, even though the scores are reported as the kappa value to compensate for different chance accuracies, the robot still performs better when given fewer options to pick from than when given more. Conversely, Figure 6.12b shows that when the number of patterns present in the matrix increases, the robot gets better at solving the task. Interestingly, even though Figure 6.12b was computed using the context distances method without supervision (as opposed to the category distances method with supervision as in all the other figures[5]), it was still possible to achieve 100% accuracy on matrices with 6 patterns when the maximum possible over all tasks was 44.6%. Intuitively this makes sense because the more patterns present in a matrix, the more constrained the possible candidate objects are, and thus the easier the task is to solve.

Figure 6.12c shows the robot's performance when the objective function was computed only across the rows of the matrix in each task, down the columns, or both. As expected, the robot is able to perform better when using an objective function that takes into account the information across the rows and down the columns. Also as expected, the robot's performance when only using rows or only using columns is approximately the same. This is likely due to the fact that the task generation algorithm treats rows and columns identically.

Figures 6.12d, 6.12e, and 6.12f show the robot's performance on different subsets of the

[5] This was done because in the version of this graph that used the category distances method with supervision, all the lines performed maximally well, making it impossible to perceive any difference in performance.

matrix completion tasks. Figure 6.12d shows the performance on tasks that include at least one instance of each of the different rules; Figure 6.12e shows the robot's performance on tasks that contain each of the different patterns; and Figure 6.12f shows the robot's performance on tasks including at least one pattern over each of the three different properties of the objects.

Figure 6.12e shows that the robot performed better on tasks in which the matrix had at least one pattern that was over the property *weight*. This is confirmed by Figure 6.12f, which shows the line for *weight* to be higher than the other two. Figure 6.12d even shows that the robot performs better when the rules *increment* and *decrement* are included, which, as stated in Section 6.4.2, are applied exclusively to *weight*. This indicates that, overall, the robot performed better on tasks that required it to perceive the weight of the objects. Intuitively this makes sense because for many of the behaviors, the robot was supporting or moving the full weight of the object, meaning the data collected from the proprioceptive modality often contained information about the weight of the object. Conversely, only a few of the behaviors caused the contents of the objects to shift and register a sound that the robot could detect with its microphones, and only one behavior (*look*) was used to extract color features. Thus, as the number of contexts available to the robot is increased, it is more likely that a context that can reliably perceive *weight* will be selected, which would improve the robot's performance on tasks that involve *weight*.

Additionally, the results shown in Figure 6.12 suggest that the robot tends to perform better when the task is more constrained (either in the form of fewer candidate objects to choose from or more patterns present in the matrix). While this was not entirely unexpected, it was surprising to find that even the worst performing distance function method (context distances without supervision) was able to achieve 100% accuracy on tasks with 6 patterns when given enough contexts.

Figure 6.13 shows the number of matrix completion tasks for three different types of difficulty. It is worthwhile to note that in Figure 6.13b the counts for the different rule types are far from uniformly distributed, and even in Figure 6.13a the counts are not uniform. This is due to the interdependencies between patterns. For example, the only way for a 3 × 3 matrix to have the *increment:weight* pattern present across the rows is for the first object in each row to be *light*, the second to be *medium*, and the third to be *heavy*. Since every row must have those values in order for *increment:weight* to be present across the rows, then that necessarily implies that all the weights are constant down each column, or that the pattern *constant:weight* is always present in the columns when *increment:weight* is present in the rows.

There are many other interdependencies between the patterns. This is illustrated in Figure 6.13a, which shows that, despite the fact that the task generation algorithm only selects 2 patterns when generating tasks, most tasks have more than 2 patterns. Also, Figure 6.13b shows that the *constant* and *permutation* rules tend to appear much more frequently in tasks than the *increment* and *decrement* rules. These interdependencies often mean that matrix completion tasks have redundant information. This is not the case for the properties of the objects, though, as Figure 6.13c shows that the distribution over tasks that include at least one pattern for each property is approximately uniform.

6.6 Summary

In this chapter we used the framework described in Chapter 1 to solve matrix completion tasks. The robot was tested on matrix completion tasks composed of objects that varied by *contents*, *color*, and *weight*. It was able to gather information about the objects by interacting with them while simultaneously recording from multiple sensory modalities. We then generated a set of 500 matrix completion tasks using those objects and posed them to the robot. Using all 21 sensorimotor contexts and the best distance method, it was able to achieve 99.4% accuracy on the set of tasks. That is, it was able to pick the correct answer for all but 3 of the 500 tasks.

The tasks posed in this chapter utilized objects that varied by multiple, independent properties. Because of this, the robot was required to synthesize information from multiple contexts in order to solve the tasks. In Chapters 4 and 5, we found that when a given set of tasks utilized objects that varied largely by a single property, a single, well-picked context was sufficient to solve those tasks with a high degree of accuracy. In this chapter we extended that framework and showed that it can be used to solve tasks that require the perception of multiple properties.

Overall, the robot was able to successfully solve a variety of matrix completion tasks using grounded, sensorimotor information. In previous work, it has been shown that robots can use exploratory behaviors to solve tasks such as object recognition (Sinapov et al., 2011a) and odd-one-out (Sinapov and Stoytchev, 2010b). This chapter showed that exploratory behaviors also work well for solving matrix completion tasks. This shows that robots that use exploratory behaviors and ground their knowledge in their own sensorimotor contexts can not only perceive useful information about objects, but also can use that information to solve a variety of tasks.

CHAPTER 7. SUMMARY, CONCLUSION, AND FUTURE WORK

7.1 Thesis Summary

In this thesis we investigated the research question “How can a robot solve multi-object perceptual reasoning tasks using embodied representations of the objects?” Chapter 4 investigated how a robot can solve the object pairing task. In those experiments, the tasks were posed to the robot using standard Montessori objects. The robot first explored the 4 sets of objects with its 10 exploratory behaviors. The objects in each set varied by one specific property but were identical in all other ways (e.g., the *weight cylinders* varied by weight but were otherwise identical). During each interaction with each object, the robot recorded from both its audio and proprioceptive modalities. It then used this information to determine the perceptual similarity between each pair of objects. Given this, the robot attempted to pair the objects within each set. The robot was able to successfully pair the objects with a high degree of accuracy.

Chapter 5 extended the work done in Chapter 4. The robot’s task in those experiments was to complete the ordering in a group of objects. More formally, given an ordered set of objects and an unordered set of objects, the robot had to select the object from the unordered set that best completed the ordering in the ordered set. The robot first interacted with 3 sets of objects by performing 10 exploratory behaviors on each object. In this case, the robot computed distance scores between every pair of objects (rather than similarity scores as before, though they are analogous concepts). We then posed 150 order completion tasks to the robot (3 sets × 50 tasks per set). The robot used the computed distance scores to measure how well each candidate object completed each ordering, and then selected the best one. The robot was able to solve the order completion task with a high degree of accuracy. We also found that by using boosting, the robot could improve its performance over just simply performing weighted voting between different information sources.

Chapter 6 investigated the robot's ability to solve a different type of perceptual reasoning task. In those experiments, the robot attempted to solve the matrix completion task. The robot was presented with a set of objects arranged in a grid, with the lower-right object missing, and with a set of candidate objects. The robot had to pick the object from the set of candidates that best fit in the missing spot in the grid. Like in the previous two experiments, the robot first interacted with the set of objects by performing 10 exploratory behaviors on each object. Once again, the robot computed the perceptual distances between every pair of objects. We then posed 500 randomly generated matrix reasoning tasks to the robot. The robot used the computed distances to see which candidate object best fit with the patterns that it detected in the matrix. The robot was able to solve the matrix completion tasks significantly better than chance, getting only 3 of the 500 tasks wrong when using the best methodology. We also found that when the robot used distance measures based on category features rather than raw features and when it used some supervision to refine the distance measures, as opposed to being completely unsupervised, it was able to perform better.

7.2 Conclusion

The goal of this thesis was to investigate the ability of robots to solve tasks from intelligence tests. In Chapter 1 we noted that only perceptual reasoning tasks are both well-suited and feasible for robots to currently solve (e.g., because they do not require the robot to understand language). Chapter 1 also introduced a framework for solving perceptual reasoning tasks and in Chapters 4, 5, and 6 this framework was used to solve three different tasks. Sinapov and Stoytchev (2010b) had already shown that this framework works well for solving the odd-one-out task, and this thesis extended it to work for solving the object pairing task, the order completion task, and the matrix completion task. While this isn't an exhaustive list of all possible perceptual reasoning tasks, it does strongly suggest that this framework does in fact generalize to many perceptual reasoning tasks. This implies that the embodied approach to robotics can be very useful for performing many of the tasks on intelligence tests, and by extension many real-world tasks that are related to tasks on intelligence tests.

This thesis showed that a robot, using multiple sensory modalities in combination with multiple exploratory behaviors, can solve perceptual reasoning tasks with a high degree of accuracy. Interestingly we found that certain modalities in combination with certain behaviors were better for some specific tasks than other combinations. More specifically, some sensorimotor contexts were better at perceiving certain properties than others (e.g., *proprioception-lift* was better at perceiving weight). This meant that for tasks that only required the perception of one object property to solve (e.g., ordering objects exclusively by weight), contexts that could perceive that property performed near perfectly while others performed poorly. However, we found that no individual context could perform well for tasks that required perception of multiple properties in order to solve (e.g., matrix completion tasks). Instead, the robot had to synthesize the information from multiple contexts in order to achieve success. This shows that as the tasks become more complex, it becomes necessary for the robot to utilize multiple, heterogeneous sources of information and to learn how to usefully combine them.

It is worthwhile to note that, in all experiments described in this thesis, the best performance was always achieved using a limited amount of supervision. More specifically, the robot was able to solve the tasks with a high degree of accuracy when only given training data in the form of example tasks with correct answers. This is similar to the methodology of the Montessori style of education, where the activities are often self-directed as opposed to taught by an instructor (Montessori, 1912; Lillard, 2008; Lillard and Else-Quest, 2006). For example, each of the Montessori sound cylinders has a colored dot on its base so that, after a student has finished pairing them, he or she can flip the objects over to verify the solution. Similarly, in this thesis, we gave the robot example tasks with correct solutions so that it could "verify" its own work in order to improve its performance. This indicates that limited, task-specific supervision can be very useful for solving perceptual reasoning tasks.

While conducting the research described in this thesis we encountered many challenges. One of the primary challenges was due to our desire to use the same set of exploratory behaviors that had been used in previous work (Sinapov et al., 2008, 2009; Bergquist et al., 2009; Sinapov et al., 2013). With the exception of adding the *rattle* behavior, we did not modify the behavioral repertoire of the robot from the immediately previous work (Sinapov et al., 2013) because we wanted to verify that it was applicable to a wide variety of tasks. We were able to show that this set of behaviors does indeed work on many different tasks. This, we believe, highlights one of the major strengths of the developmental apporach to robotics as it shows that a common set of embodied representations that are grounded in multiple sensorimotor contexts can be used to solve a broad set of tasks. Traditional approaches focus on specific solutions for specific tasks and as a result have problems with generalization to even slightly different tasks.

Another challenge we encountered was developing the objects for the tasks, particularly for the matrix completion task. On intelligence tests, images are most commonly used as the objects in the various tasks. However, in order to pose the tasks to the robot, we needed to develop physical objects for the robot to manipulate. For each of the three tasks, we were able to use objects that maintained the underlying structure of the problems while moving to the domain of physical objects. We showed that if a robot can build an understanding of the objects, then it can successfully solve various tasks.

The specific perceptual reasoning tasks presented to the robot in this thesis gradually increased in complexity and built on each other. The first task showed that, using exploratory behaviors, a robot can determine the property by which a set of objects varies and then match those objects based on that property. The second task showed that, once a robot has detected how a set of objects varies, it can then reason about that variation in order to solve the task. Finally, the third task showed that a robot can detect and reason about multiple properties by which a set of objects varies at the same time. Over the course of this thesis, the robot's ability to solve perceptual reasoning tasks progressed from being able to solve relatively simple tasks to solving more complex tasks.

7.3 Future Work

In this thesis we showed that a robot can solve a variety of perceptual reasoning tasks. Future work could extend this to other perceptual reasoning tasks, such as sequence completion (i.e., given a sequence of objects, pick an object that completes it based on the patterns present in the sequence) or block design (i.e., given some differently colored and shaped blocks, assemble them into a predefined pattern). Given that the framework described in this thesis for solving perceptual reasoning tasks has been shown to work for multiple tasks, it is reasonable to expect that it could be used to successfully solve other perceptual reasoning tasks as well.

It would also be interesting to solve other types of tasks from intelligence tests with robots, such as verbal reasoning tasks. As stated in Chapter 1, however, these types of tasks would require that the robot be able to learn large amounts of background knowledge, such as language. For example, in order to solve word relation tasks, a robot would have to understand the meaning of various words and how they relate to each other. Nonetheless, it would be interesting to develop robotic systems capable of learning this type of knowledge and to test those systems using the same type of tasks designed to test humans.

Another possible direction for future work is an improved objective function. In this thesis, all the tasks required their own, task-specific objective function. Future work could investigate methods for creating a generalized objective function that would work across a wide variety of perceptual reasoning tasks. This would require a unified method for posing tasks and it would require the robot to be able to learn the properties of each task. Though difficult, doing so would allow robots to solve many more perceptual reasoning tasks without requiring the development of task-specific methodology.

BIBLIOGRAPHY

Amant, R. S. and Wood, A. B. (2005). Tool use for autonomous agents. In *Proceedings of the National Conference on Artificial Intelligence (AAAI)*, pages 184–189, Pittsburgh, PA.

Bergquist, T., Schenck, C., Ohiri, U., Sinapov, J., Griffith, S., and Stoytchev, A. (2009). Interactive object recognition using proprioceptive feedback. In *Proceedings of the IROS Workshop: Semantic Perception for Robot Manipulation*, St. Lousi, MO.

Campbell, M., Hoane Jr, A. J., and Hsu, F.-h. (2002). Deep blue. *Artificial intelligence*, 134(1):57–83.

Carpenter, P. A., Just, M. A., and Shell, P. (1990). What one intelligence test measures: a theoretical account of the processing in the Raven progressive matrices test. *Psychological review*, 97(3):404–431.

Carroll, O. (1997). The three-stratum theory of cognitive abilities. *Contemporary Intellectual Assessment: Theories, Tests, and Issues*, pages 122–130.

Caruso, D. A. (1993). Dimensions of quality in infants' exploratory behavior: Relationships to problem-solving ability. *Infant Behavior and Development*, 16(4):441–454.

Chan, A. and Pampalk, E. (2002). Growing hierarchical self organising map (GHSOM) toolbox: visualisations and enhancements. In *Proceedings of the 9th International Conference on Neural Information Processing*, volume 5, pages 2537–2541, Singapore.

Chapelle, O., Chang, Y., and Liu, T. (2011). Future directions in learning to rank. In *Journal of Machine Learning Research: Workshop and Conference Proceedings*, volume 14, pages 91–100.

Cirillo, S. (2010). An anthropomorphic solver for Raven's progressive matrices. Master's thesis, Chalmers University of Technology, Göteborg, Sweden.

Cohen, J. (1960). A coefcient of agreement for nominal scales. *Educational and Psychological Measurement*, 20(1):37–46.

Cohen, R. J., Swerdlik, M. E., and Phillips, S. M. (1999). *Psychological testing and assessment.* Mayfield, Mountain View, CA.

Daehler, M., Lonardo, R., and Bukatko, D. (1979). Matching and equivalence judgments in very young children. *Child Development*, 50(1):170–179.

Darwin, C. (1874). *The Descent of Man and Selection in Relation to Sex.* Appleton, NY.

Deary, I. J., Strand, S., Smith, P., and Fernandes, C. (2007). Intelligence and educational achievement. *Intelligence*, 35(1):13–21.

DuBois, P. H. (1970). *A history of psychological testing.* Allyn and Bacon, Boston, MA.

Dugbartey, A. T., Sanchez, P. N., Rosenbaum, J. G., Mahurin, R. K., Davis, J. M., and Townes, B. D. (1999). WAIS-III matrix reasoning test performance in a mixed clinical sample. *The Clinical Neuropsychologist*, 13(4):396–404.

Ebeling, K. and Gelman, S. (1988). Coordination of size standards by young children. *Child Development*, 59(4):888–896.

Ebeling, K. and Gelman, S. (1994). Children's use of context in interpreting big and little. *Child Development*, 65(4):1178–1192.

Endres, F., Plagemann, C., Stachniss, C., and Burgard, W. (2009). Unsupervised discovery of object classes from range data using latent Dirichlet allocation. In *Proceedings of Robotics: Science and Systems*, Seattle, WA.

Ferrucci, D., Brown, E., Chu-Carroll, J., Fan, J., Gondek, D., Kalyanpur, A. A., Lally, A., Murdock, J. W., Nyberg, E., Prager, J., et al. (2010). Building Watson: An overview of the DeepQA project. *AI magazine*, 31(3):59–79.

Fitzpatrick, P., Metta, G., Natale, L., Rao, S., and Sandini, G. (2003). Learning about objects through action-initial steps towards artificial cognition. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, volume 3, pages 3140–3145, Taipei, Taiwan.

Freund, Y. and Schapire, R. (1995). A desicion-theoretic generalization of on-line learning and an application to boosting. *Computational Learning Theory*, 904:23–37.

Fuchs, L. S., Fuchs, D., Stuebing, K., Fletcher, J. M., Hamlett, C. L., and Lambert, W. (2008). Problem solving and computational skill: Are they shared or distinct aspects of mathematical cognition? *Journal of Educational Psychology*, 100(1):30–47.

Gaillard, M. K., Grannis, P. D., and Sciulli, F. J. (1999). The standard model of particle physics. *Reviews of Modern Physics*, 71(2):S96–S111.

Gibson, E. J. (1988). Exploratory behavior in the development of perceiving, acting, and the acquiring of knowledge. *Annual review of psychology*, 39(1):1–42.

Glickman, S. E. and Sroges, R. W. (1966). Curiosity in zoo animals. *Behaviour*, 26(1):151–188.

Graham, F., Ernhart, C., Craft, M., and Berman, P. (1964). Learning of relative and absolute size concepts in preschool children. *Journal of Experimental Child Psychology*, 1(1):26–36.

Gregson, D. (1989). Program notes. *The Westgate-Mainly Mozart Festival, Under the Stars at the Old Globe, San Diego, CA*, page 24.

Griffith, S., Sinapov, J., Miller, M., and Stoytchev, A. (2009). Toward interactive learning of object categories by a robot: A case study with container and non-container objects. In *Proceedings of the 8th IEEE International Conference on Development and Learning (ICDL)*, pages 1–6, Shanghai, China.

Griffith, S., Sinapov, J., and Stoytchev, A. (2008). Toward learning to detect and use containers. Poster abstract at the 7th IEEE International Conference on Development and Learning (ICDL), Monterey, CA.

Griffith, S., Sinapov, J., Sukhoy, V., and Stoytchev, A. (2010). How to separate containers from non-containers? A behavior-grounded approach to acoustic object categorization. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 1852–1859, Anchorage, AK.

Griffith, S., Sinapov, J., Sukhoy, V., and Stoytchev, A. (2012a). A behavior-grounded approach to forming object categories: Separating containers from noncontainers. *IEEE Transactions on Autonomous Mental Development*, 4(1):54–69.

Griffith, S. and Stoytchev, A. (2010). Interactive categorization of containers and non-containers by unifying categorizations derived from multiple exploratory behaviors. In *Proceedings of the 24-th National Conference on Artificial Intelligence (AAAI)*, pages 11–15, Atlanta, GA.

Griffith, S., Sukhoy, V., and Stoytchev, A. (2011). Using sequences of movement dependency graphs to form object categories. In *Proceedings of the 11th IEEE-RAS International Conference on Humanoid Robots (Humanoids)*, pages 715–720, Bled, Slovenia.

Griffith, S., Sukhoy, V., Wegter, T., and Stoytchev, A. (2012b). Object categorization in the sink: Learning behavior–grounded object categories with water. In *Proceedings of the 2012 ICRA Workshop on Semantic Perception, Mapping and Exploration*, St. Paul, MN.

Hagmann-von Arx, P., Meyer, C., and Grob, A. (2008). Assessing intellectual giftedness with the WISC-IV and the IDS. *Zeitschrift für Psychologie*, 216(3):172–179.

Hayashi, M. and Matsuzawa, T. (2003). Cognitive development in object manipulation by infant chimpanzees. *Animal Cognition*, 6(4):225–233.

Hayashi, M., Takeshita, H., and Matsuzawa, T. (2006). Cognitive development in apes and humans assessed by object manipulation. In *Cognitive Development in Chimpanzees*, pages 395–410. Springer.

Henderson, B. B. and Wilson, S. E. (1991). Intelligence and curiosity in preschool children. *Journal of School Psychology*, 29(2):167–175.

Hernández-Orallo, J. and Dowe, D. L. (2010). Measuring universal intelligence: Towards an anytime intelligence test. *Artificial Intelligence*, 174(18):1508–1539.

Horn, J. L. and Cattell, R. B. (1966). Refinement and test of the theory of fluid and crystallized intelligence. *Journal of Educational Psychology*, 57:253–270.

Hunter, J. E. and Schmidt, F. L. (1998). The validity and utility of selection methods in personnel psychology: Practical and theoretical implications of 85 years of research findings. *Psychological bulletin*, 124(2):262–274.

Inglis, I. and Shepherd, D. (1994). Rats work for food they then reject: Support for the information-primacy approach to learned industriousness. *Ethology*, 98(2):154–164.

Insa-Cabrera, J., Dowe, D. L., España-Cubillo, S., Hernández-Lloreda, M. V., and Hernández-Orallo, J. (2011). Comparing humans and AI agents. In *Proceedings of the 4th International Conference on Artificial General Intelligence (AGI)*, volume 6830, pages 122–132, Mountain View, CA.

Kaufman, A. (1994). *Intelligent Testing With the WISC-III.* John Wiley & Sons, New York.

Kaufman, A. and Kaufman, N. (1983). *Kaufman Assessment Battery for Children.* American Guidance Service, Circle Pines, MN.

Kaufman, A. S. (2004). Kaufman assessment battery for children, (KABC-II). *Circle Pines, MN: AGS Publishing.*

Kaufman, A. S. (2009). *IQ testing 101.* Springer Publishing Company, New York, NY.

Kinnaman, A. (1902). Mental life of two macacus rhesus monkeys in captivity. *The American Journal of Psychology*, 13(1):98–148.

Krotkov, E., Klatzky, R., and Zumel, N. (1997). Robotic perception of material: Experiments with shape-invariant acoustic measures of material type. In *Experimental Robotics IV*, volume 223, pages 204–211.

Kunda, M., McGreggor, K., and Goel, A. (2010). Taking a look (literally!) at the Ravens intelligence test: Two visual solution strategies. In *Proceedings of the 32nd Annual Meeting of the Cognitive Science Society*, Portlan, OR.

Lederman, S. J. and Klatzky, R. L. (1987). Hand movements: A window into haptic object recognition. *Cognitive psychology*, 19(3):342–368.

Lee, K., Hon, H., and Reddy, R. (1990). An overview of the SPHINX speech recognition system. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 38(1):35–45.

Legg, S. and Hutter, M. (2007). Universal intelligence: A definition of machine intelligence. *Minds and Machines*, 17(4):391–444.

Leslie, A. and Chen, M. (2007). Individuation of pairs of objects in infancy. *Developmental Science*, 10(4):423.

Lillard, A. (2008). *Montessori: The Science Behind the Genius*. Oxford University Press, USA.

Lillard, A. and Else-Quest, N. (2006). The early years: Evaluating Montessori. *Science*, 313(5795):1893–1894.

Lincoln, D. (2012). The inner life of quarks. *Scientific American*, 307(5):36–43.

Little, D., Lewandowsky, S., and Griffiths, T. (2012). A Bayesian model of rule induction in Raven's progressive matrices. In *Proceedings of the 34th Annual Conference of the Cognitive Science Society*, pages 1918–1923, Sapporo, Japan.

Littman, M. L., Keim, G. A., and Shazeer, N. (2002). A probabilistic approach to solving crossword puzzles. *Artificial Intelligence*, 134(1):23–55.

Lorenz, K. (1996). *Learning as Self-Organization*, chapter Innate Bases of Learning. Lawrence Erlbaum Pub., Mahwah, NJ.

Lovett, A., Forbus, K., and Usher, J. (2010). A structure-mapping model of Ravens progressive matrices. In *Proceedings of the 32nd Annual Meeting of the Cognitive Science Society*, volume 10, pages 2761–2766, Portland, OR.

Markoff, J. (2011). Computer wins on Jeopardy!: Trivial, its not. *New York Times*, 16 February, 16.

McPherson, W. and Holcomb, P. (1999). An electrophysiological investigation of semantic priming with pictures of real objects. *Psychophysiology*, 36(01):53–65.

Messer, D. J., McCarthy, M. E., McQuiston, S., MacTurk, R. H., Yarrow, L. J., and Vietze, P. M. (1986). Relation between mastery behavior in infancy and competence in early childhood. *Developmental Psychology*, 22(3):366–372.

Messer, D. J., Rachford, D., McCarthy, M., and Yarrow, L. (1987). Assessment of mastery behavior at 30 months: Analysis of task-directed activities. *Developmental Psychology*, 23(6):771.

Montessori, M. (1912). *The Montessori Method.* Frederick A. Stokes Co., New York.

Naglieri, J. A. and Bornstein, B. T. (2003). Intelligence and achievement: Just how correlated are they? *Journal of Psychoeducational Assessment*, 21(3):244–260.

Nakamura, T., Nagai, T., and Iwahashi, N. (2007). Multimodal object categorization by a robot. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2415–2420, San Diego, CA.

Natale, L., Metta, G., and Sandini, G. (2004). Learning haptic representation of objects. In *Proceedings of the International Conference on Intelligent Manipulation and Grasping.*

Navarro, G. (2001). A guided tour to approximate string matching. *ACM computing surveys (CSUR)*, 33(1):31–88.

Neisser, U., Boodoo, G., Bouchard Jr, T. J., Boykin, A. W., Brody, N., Ceci, S. J., Halpern, D. F., Loehlin, J. C., Perloff, R., Sternberg, R. J., et al. (1996). Intelligence: Knowns and unknowns. *American psychologist*, 51(2):77.

Ng, A. Y., Jordan, M. I., Weiss, Y., et al. (2002). On spectral clustering: Analysis and an algorithm. *Advances in neural information processing systems*, 2:849–856.

Nolfi, S. and Marocco, D. (2002). Active perception: A sensorimotor account of object categorization. In *From Animals to Animats: 7*, pages 266–271.

Pedersen, F. A. and Wender, P. H. (1968). Early social correlates of cognitive functioning in six-year-old boys. *Child development*, pages 185–193.

Piaget, J. (1952). *The Origins of Intelligence in Children.* International Universities Press, New York.

Pitamic, M. (2004). *Teach Me to Do It Myself.* Elwin Street Productions.

Power, T. (2000). *Play And Exploration in Children And Animals.* Lawrence Erlbaum, Mahwah, NJ.

Prabhakaran, V., Smith, J. A., Desmond, J. E., Glover, G. H., Gabrieli, J. D., et al. (1997). Neural substrates of fluid reasoning: an FMRI study of neocortical activation during performance of the Raven's progressive matrices test. *Cognitive psychology*, 33:43–63.

Rasmussen, D. and Eliasmith, C. (2011). A neural model of rule generation in inductive reasoning. *Topics in Cognitive Science*, 3(1):140–153.

Rathunde, K. and Csikszentmihalyi, M. (2005). Middle school students motivation and quality of experience: A comparison of Montessori and traditional school environments. *American Journal of Education*, 111(3):341–371.

Raven, J. (2000). The Raven's progressive matrices: Change and stability over culture and time. *Cognitive psychology*, 41(1):1–48.

Raven, J. C. (1938). *Progressive matrices.* Éditions scientifiques et psychotechniques.

Richardson, K. (1991). Reasoning with ravenin and out of context. *British Journal of Educational Psychology*, 61(2):129–138.

Romanes, G. J. (1882). *Animal intelligence.* Kegan Paul, Trench & Co, London, Great Britain.

Rooks, B. (2006). The harmonious robot. *Industrial Robot: An International Journal*, 33(2):125–130.

Ruff, H. A. and Dubiner, K. (1987). Stability of individual differences in infants'manipulation and exploration of objects. *Perceptual and Motor Skills*, 64(3c):1095–1101.

Saenko, K. and Darrell, T. (2008). Object category recognition using probabilistic fusion of speech and image classifiers. In *Proceedings of the 4th International Workshop Machine Learning for Multimodal Interaction, Revised Selected Papers*, volume 4892, pages 36–47, Brno, Czech Republic.

Sanghi, P. and Dowe, D. L. (2003). A computer program capable of passing IQ tests. In *Proceedings of the 4th International Conference on Cognitive Science (ICCS)*, pages 570–575.

Sato, Y. and Inoue, H. (2010). Solving sudoku with genetic operations that preserve building blocks. In *Proceedings of the IEEE Symposium on Computational Intelligence and Games (CIG)*, pages 23–29, Copenhagen, Denmark.

Sattler, J. M. (2008). *Assessment of children: Cognitive foundations*. JM Sattler, San Diego, CA.

Scerri, E. R. (2011). *The Periodic Table: A Very Short Introduction*. Oxford University Press, Oxford.

Schaeffer, J., Lake, R., Lu, P., and Bryant, M. (1996). CHINOOK: The world man-machine checkers champion. *AI Magazine*, 17(1):21.

Sharp, S. E. (1899). Individual psychology: A study in psychological method. *The American Journal of Psychology*, 10(3):329–391.

Sinapov, J., Bergquist, T., Schenck, C., Ohiri, U., Griffith, S., and Stoytchev, A. (2011a). Interactive object recognition using proprioceptive and auditory feedback. *The International Journal of Robotics Research*, 30(10):1250–1262.

Sinapov, J., Schenck, C., Staley, K., Sukhoy, V., and Stoytchev, A. (2013). Grounding semantic categories in behavioral interactions: Experiments with 100 objects. *Robotics and Autonomous Systems (to appear)*.

Sinapov, J. and Stoytchev, A. (2008). Detecting the functional similarities between tools using a hierarchical representation of outcomes. In *Proceedings of the 7th IEEE International Conference on Development and Learning (ICDL)*, pages 91–96, Monterey, CA.

Sinapov, J. and Stoytchev, A. (2010a). The boosting effect of exploratory behaviors. In *Proceedings of the 24th National Conference on Artificial Intelligence (AAAI)*, pages 1613–1618, Atlanta, GA.

Sinapov, J. and Stoytchev, A. (2010b). The odd one out task: Toward an intelligence test for robots. In *Proceedings of the 9th IEEE International Conference on Development and Learning (ICDL)*, pages 126–131, Ann Arbor, MI.

Sinapov, J. and Stoytchev, A. (2011). Object category recognition by a humanoid robot using behavior-grounded relational learning. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 184–190, Shanghai, China.

Sinapov, J., Sukhoy, V., Sahai, R., and Stoytchev, A. (2011b). Vibrotactile recognition and categorization of surfaces by a humanoid robot. *IEEE Transactions on Robotics*, 27(3):488–497.

Sinapov, J., Wiemer, M., and Stoytchev, A. (2008). Interactive learning of the acoustic properties of objects by a robot. In *Procceedings of the RSS Workshop on Robot Manipulation: Intelligence in Human Environments, Zurich, Switzerland*.

Sinapov, J., Wiemer, M., and Stoytchev, A. (2009). Interactive learning of the acoustic properties of household objects. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 2518–2524, Kobe, Japan.

Slonaker, J. R. (1912). The normal activity of the albino rat from birth to natural death, its rate of growth and the duration of life. *Journal of Animal Behavior*, 2:20–42.

Small, W. S. (1899). Notes on the psychic development of the young white rat. *The American Journal of Psychology*, 11(1):80–100.

Spearman, C. (1904). "General Intelligence," Objectively determined and measured. *The American Journal of Psychology*, 15(2):201–292.

Spearman, C. (1914). The theory of two factors. *Psychological Review*, 21(2):101.

Stoytchev, A. (2005). Behavior-grounded representation of tool affordances. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 3060–3065, Barcelona, Spain.

Sugarman, S. (1981). The cognitive basis of classification in very young children: An analysis of object-ordering trends. *Child Development*, 52(4):1172–1178.

Sun, J., Moore, J., Bobick, A., and Rehg, J. (2010). Learning visual object categories for robot affordance prediction. *The International Journal of Robotics Research*, 29(2-3):174.

Takamuku, S., Hosoda, K., and Asada, M. (2008). Object category acquisition by dynamic touch. *Advanced Robotics*, 22(10):1143–1154.

Tenenbaum, J., De Silva, V., and Langford, J. (2000). A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500):2319–2323.

Terman, L. M. (1916). *The Measurement of intelligence*. Houghton Mifflin, Boston, MA.

Torres-Jara, E., Natale, L., and Fitzpatrick, P. (2005). Tapping into touch. In *Proceedings of the Fifth International Workshop on Epigenetic Robotics*, Osaka, Japan.

Turing, A. M. (1950). Computing machinery and intelligence. *Mind*, 59(236):433–460.

Turvey, M. T. (1996). Dynamic touch. *American Psychologist*, 51(11):1134.

Vauclair, J. and Bard, K. A. (1983). Development of manipulations with objects in ape and human infants. *Journal of Human Evolution*, 12(7):631–645.

Verbeek, M. E., Drent, P. J., and Wiepkema, P. R. (1994). Consistent individual differences in early exploratory behaviour of male great tits. *Animal Behaviour*, 48(5):1113–1121.

Watt, D. C. (1998). Lionel Penrose, FRS (1898–1972) and eugenics: Part one. *Notes and Records of the Royal Society of London*, 52(1):137–151.

Wechsler, D. (1939). *The measurement of adult intelligence.* Williams & Wilkins Co, Baltimore.

Wechsler, D. (1997). *Wechsler Adult Intelligence Scale–third edition: Administration and scoring manual.* Psychological Corporation, San Antonio, TX.

Wechsler, D. (2003). Wechsler Intelligence Scale for Children–fourth edition (WISC-IV). *San Antonio, TX: The Psychological Corporation.*

Weisler, A. and McCall, R. B. (1976). Exploration and play: Resume and redirection. *American Psychologist*, 31:492–508.

Westergaard, G. C. (1992). Object manipulation and the use of tools by infant baboons (Papio cynocephalus anubis). *Journal of Comparative Psychology*, 106(4):398–403.

Westergaard, G. C. (1993). Development of combinatorial manipulation in infant baboons (Papio cynocephalus anubis). *Journal of comparative psychology*, 107(1):34–38.

Wissler, C. (1901). The correlation of mental and physical tests. *Psychological Monographs: General and Applied*, 3(6):1–62.

Yarrow, L. J., McQuiston, S., MacTurk, R. H., McCarthy, M. E., Klein, R. P., and Vietze, P. M. (1983). Assessment of mastery motivation during the first year of life: Contemporaneous and cross-age relationships. *Developmental Psychology*, 19(2):159.

Younger, B. (1985). The segregation of items into categories by ten-month-old infants. *Child Development*, 56(6):1574–1583.