Interactive Learning of the Acoustic Properties of Objects by a Robot

Jivko Sinapov, Mark Wiemer, and Alexander Stoytchev
Developmental Robotics Lab
Iowa State University
{jsinapov, banff, alexs}@iastate.edu

***Abstract*— Humans can perceive various object properties based solely on the sounds that the objects make when an action is performed on them. Similarly, robots in human-inhabited environments must be capable of learning and reasoning about the acoustic properties of the objects with which they interact. Such an ability would allow a robot to infer some object properties even if the object is not in direct line of sight. This paper presents a framework that allows a robot to infer the object with which it is interacting from the sounds generated by the object during the interaction. The framework is evaluated on a 7-DOF Barrett WAM robot which performs pushing, grasping, and dropping behaviors on 18 different objects. The results show that the robot is able to accurately recognize objects (e.g., bottles, cups, balls, etc.) based on their acoustic properties. Furthermore, the recognition accuracy can be improved if the robot performs a combination of different exploratory behaviors on each object.**

I. Introduction

Human beings have the remarkable ability to extract the physical properties of objects from the sounds that they produce [1, 2]. Unlike our sense of vision, which is always constrained to a particular viewing direction, our auditory sense allows us to infer events in the world that are often outside the reach or range of other sensory modalities [3].

Using sound as a source of information would undoubtedly help a robot detect and reason about events in a human-inhabited environment. For example, if a robot accidentally knocks over an object that is outside of its field of view, the sound generated by the object as it falls to the ground will be the only source of information about the nature of the object. A robot grasping an object out of sight (e.g., a toy in a box) will only have access to auditory and tactile information regarding the type of object it is interacting with. Similarly, if a human interacts with an object that is outside the robot's field of view, the robot can use the detected sounds to infer the nature of the object and the type of interaction. These types of situations clearly present a challenge to traditional object recognition frameworks which rely heavily on computer vision methods.

This paper addresses the problem of how a robot can recognize the object it is interacting with based on the detected sounds produced by the object. We present a framework in which the robot learns compact predictive models that can estimate the object class given the robot's exploratory behavior and the resulting sounds. Three different algorithms representing distinct families of machine learning methods are evaluated: k-Nearest Neighbor (an *instance-based* method), Support Vector Machine (a *discriminative* learning algorithm), and a Bayesian Network (a *probabilistic graphical model*).

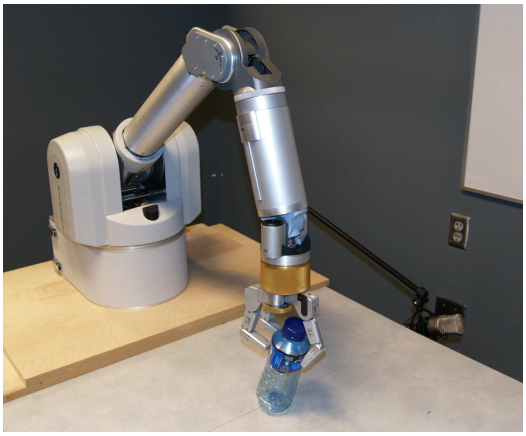


Fig. 1. The 7-DOF Barrett whole arm manipulator used in the experiments. The figure also shows the microphone used to record the sounds.

The robot used in the experiments is a 7-DOF Barrett WAM arm shown in Figure 1. The robot's behavioral repertoire consists of three different behaviors (pushing, dropping, and grasping), which it applies to all objects that it encounters. Eighteen different objects were used for performance evaluation, including a bottle, a pop can, a book, etc. The three learning algorithms were evaluated based on how well they can generalize to novel auditory data not available during the training stage. The results show that by performing a combination of behaviors, the robot is able to improve its acoustic-based object recognition performance, regardless of the type of learning algorithm that is used.

II. Related Work

Despite the vast amount of information conveyed by the acoustic properties of everyday objects, there have been relatively few studies investigating how a robot could perceive object properties using auditory information. One of the first such studies was conducted by Krotkov *et al.* [4] in which the task of the robot was to identify the material type (e.g., glass, wood, etc.) of different objects by probing them with its end effector. In that study, the robot used a hitting behavior to recognize five different materials: aluminum, brass, glass, wood, and plastic. The results indicate that the spectrogram of

the detected sound can be used as a powerful representation for discriminating between the five materials [4]. Subsequent work by Klatzky *et al.* [5] shows that modeling frequency and decay parameters of sounds can also be used to build a sound model for each material.

More recently, Richmond *et al.* [6] have proposed a robotic platform for automatic sound measurement of contact sounds. Contact sounds are defined as the sounds generated when the end effector of the robot strikes the surface of an object. In subsequent work [7], Richmond proposes modeling the spectrogram of the sounds using spectrogram averaging, in order to learn models for contact sounds induced when striking different types of materials.

Torres-Jara *et al.* [8] demonstrate how a robot can recognize objects based on the sounds they make when tapped by the robot's hand. In that study the robot performs tapping behaviors on the objects within reach and records the detected sound spectrograms. When tapping a novel object, the robot matches the spectrogram of the detected sound to one that is already in its training set which results in a prediction for the object's type. The results show that the robot is able to recognize with high accuracy four different objects of varying materials by tapping. Their work is perhaps the first example of interactive object recognition using auditory information by a robot.

Following, this paper presents a framework in which the robot uses machine learning methods in order to perform auditory object recognition of 18 different objects using 3 different behaviors. This paper also shows that by applying multiple different behaviors to an object, a robot could improve its auditory recognition performance regarding the object's type.

III. Experimental Setup

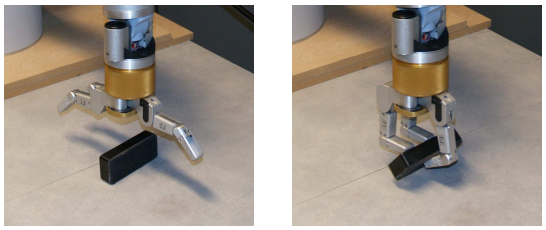
A. Robot

The robot used in the experiments is a Barrett whole arm manipulator (WAM) with the 3-fingered Barrett hand as its end effector (see Figure 1). The robot arm has 7 degrees of freedom. The hand also has 7 degrees of freedom: two per each finger, and one that controls the spread of fingers 1 and 2.

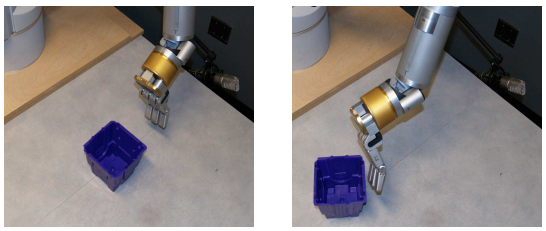
B. Exploratory Behaviors

The robot uses three exploratory behaviors (*grasp*, *push*, and *drop*) to learn the acoustic properties of different objects. The behaviors were encoded using the teach and play interface provided by the Barrett WAM API. Figure 2 shows *before* and *after* images for each of the three behaviors, which are described in more details below.

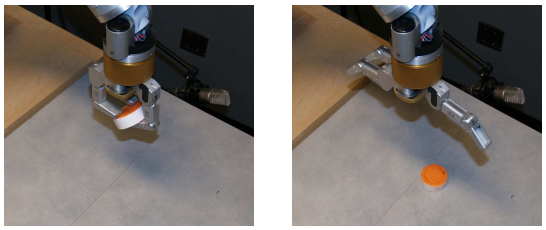
1) Grasp behavior: The object is placed in front of the robot and the Barrett hand is positioned over it with fully outstretched fingers. Next, the command to close all fingers is executed resulting in the object being grasped by the hand. Figure 2.a shows an example of a *grasp* behavior performed on a whiteboard eraser, one of the eighteen objects used in the experiments.



a) Example of a *grasp* behavior.



b) Example of a *push* behavior.



c) Example of a *drop* behavior.

Fig. 2. Examples of the *grasp*, *push*, and *drop* behaviors used by the robot.

Fig. 3. The eighteen objects used in the experiments. Top row: plastic bottle, plastic ball, rubber ball, tennis ball, plastic box, wooden plank; Second row: hockey puck, book, tin box, pop can, metal plate, soft plastic cup; Third row: paper box, eraser, metal flange, paper cup, hard plastic cup, wooden cube.

2) Push behavior: The object is placed on the table and the robot arm executes a recorded trajectory that pushes the object sideways. During this behavior, the hand is placed in an open palm configuration. An example of the *push* behavior is shown in Figure 2.b.

3) Drop behavior: The object is first grasped and then lifted to a pre-specified height above the table. Next, a command to open all three fingers is executed, resulting in the object falling and hitting the table. Figure 2.c shows the robot performing the *drop* behavior while holding the hockey puck object.

C. Objects

The set of objects, $\mathcal{O}$, that the robot interacts with consists of 18 different objects, as shown in Figure 3. The objects include different types of balls, cups, containers, a book, a bottle, a hockey puck, a whiteboard eraser, etc. The objects are made of varying materials including metal, plastic, rubber, paper, and wood. Some of the objects can be knocked down when pushed while others simply slide or roll. In addition, some of the objects bounce multiple times off the table when dropped (e.g., the three balls) while others don't.

D. Sound Recording and Feature Extraction

Sounds were recorded at a sampling rate of 44.1 KHz with 16 bit depth, processed through a Lexicon Alpha bus-powered audio interface. The audio was captured and segmented utilizing the digital audio processing package Audacity. The microphone used was a Rode NT1-A with a cardioid polar pattern having an average self noise of 5 dB. Signal leveling remained consistent for each trial while maintaining headroom to impede clipping. The microphone's output was routed to an ART Tube MP Studio microphone pre-amplifier. The pre-amplifier supplied 48 volt phantom power to the NT1-A microphone. Sufficient gain was used on the pre-amplifier to provide a suitable input level for the recording input/output device. No audio compression was used on the recordings.

During the grasping behavior, each sound is segmented such that it starts with the initiation of the grasp motor command and ends once the decibel level has dropped to that of the background noise. During the dropping behavior, each sound starts once the object hits the ground and ends once the volume level has dropped to that of the background noise. Finally, for the pushing behavior, the sound is segmented such that it starts when the hand makes its first contact with the object and ends once the dB level has returned to that of the background. For all three behaviors, the segmentation was done automatically by thresholding the dB level at the start and end of each sound.

Sound features were extracted using the log-normalized Discrete Fourier Transform (DFT) which was computed for each sound, using $2^6 + 1 = 65$ frequency bins. The SPHINX4 natural language processing library package was used to compute the DFT for each sound [9]. Next, given the DFT matrix for each sound, a 2-D histogram is computed by discretizing time into k_t bins and frequencies into k_f bins. The value for each bin in the histogram is set to the average of the values in the DFT matrix that fall into it. In all experiments conducted, k_t was set to 10 and k_f was set to 5. Hence, each sound is represented by feature vector, S, where $S \in \mathbb{R}^{5\times 10}$. Figure 4 shows an example of how the DFT of a sound is transformed into a 2-D histogram across time and frequency.

E. Data Collection

Let $\mathcal{B} = [$*grasp*, *push*, *drop*$]$ denote the set of exploratory behaviors. For each of the three behaviors, the robot performs six trials for each of the eighteen objects resulting in a total of $3 \times 6 \times 18 = 324$ recorded trials. During the i^{th} trial, the robot records a data triple of the form (B_i, O_i, S_i), where $B_i \in \mathcal{B}$ is the behavior executed, $O_i \in \mathcal{O}$ is the object on which the behavior was performed, and $S_i \in \mathbb{R}^{5\times 10}$ is the feature vector extracted from the detected sound. Each triple indicates that the sound features S_i were detected when performing behavior B_i on object O_i. Given such data, the task of the robot is to learn a model that can predict the object O_i in the interaction given the behavior B_i and sound features S_i. The next section describes the learning framework used to solve this task.

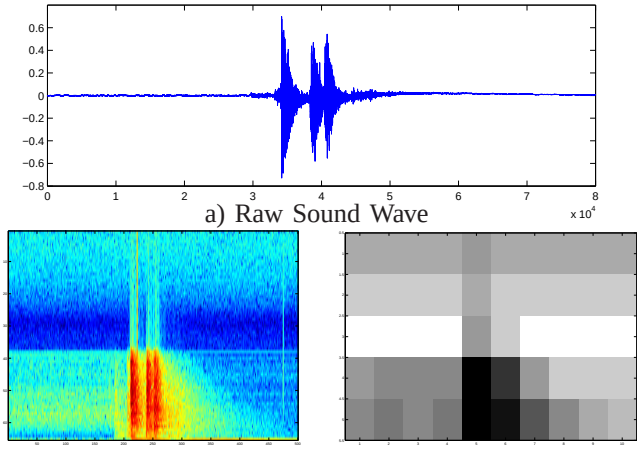


Fig. 4. Example feature extraction from the sound generated by applying the *grasp* behavior on the *pop can* object. The raw sound wave is shown in a), where the horizontal axis denotes time and the vertical axis denotes dB level. The Discrete Fourier Transform is shown in b). The resulting set of features is shown in c) as a 2-D histogram. In both b) and c) the horizontal axis denotes time, while the vertical axis denotes frequency.

IV. Learning Methodology

A. Problem Formulation

The learning task of the robot is formulated as follows: for each exploratory behavior $B \in \mathcal{B}$, learn a model M_B such that $M_B(S_i) \rightarrow O_i$, where O_i is the object present during the interaction that generated the sound feature vector S_i. In other words, the robot needs to learn a model that can predict the object class given the detected sound features. More specifically, the model should be able to estimate the conditional probability $Pr_B(O_i = o|S_i)$ for each object $o \in \mathcal{O}$, and each behavior $B \in \mathcal{B}$, given the detected sound feature vector S_i.

Given that the robot is capable of performing three different behaviors, the task is to learn the models M_{grasp}, M_{push}, and M_{drop}. For each behavior, given a set of training examples $\{S_i, O_i\}$, where $i = 1, \ldots, N$, the robot uses a supervised machine learning algorithm in order to learn a model that can estimate $Pr_B(O_i = o|S_i)$. The model can then be evaluated on novel data which was not used during the training stage. Three different machine learning algorithms are evaluated: k-Nearest Neighbor (k-NN), Support Vector Machine (SVM), and Bayesian Network. The three algorithms were chosen to represent three general families of machine learning models: *instance-based* (e.g., k-NN), *discriminative* (e.g., SVM), and *generative* (e.g., Bayesian Networks).

B. Learning Algorithms

1) K-Nearest Neighbor: K-Nearest Neighbor (k-NN) is an instance-based learning algorithm which simply stores all data points and their class labels and only uses them when the model is queried to make a prediction. The k-NN model falls within the family of *lazy learning* or *memory-based learning* algorithms [10, 11].

When asked to make a prediction on a test data point, k-NN finds the *k* closest neighbors of the query point and assigns a class label which is a smoothed average of the labels belonging to the selected neighbors. While k-NN is very simple to train and use, its performance can suffer if there are irrelevant features in the input space without the introduction of special distance and attribute weighting functions.

In the experiments conducted in this study, k was set to 3, and the WEKA [12] implementation of k-NN was used. Obtaining an estimate for $Pr_B(O_i = o|S_i)$ is done by counting the class labels of the k neighbors. For example, if two of those neighbors have class label *Rubber Ball* and one, *Tennis Ball*, then the estimated probability that the class label for the test point is *Rubber Ball* is 0.67; that of *Tennis Ball* is 0.33; and that of all other class labels is 0.0.

2) Support Vector Machine: Support Vector Machine (SVM) classifier is a supervised learning model that falls into the family of *discriminative* models [13]. Given a set of labeled inputs $(\mathbf{x}_i, y_i)_{i=1,\ldots,l}$, $\mathbf{x}_i \in \mathbb{R}^n$ and $y_i \in \{-1, +1\}$, training an SVM classifier is reduced to learning a linear decision function $f(\mathbf{x}) =< \mathbf{x}, \mathbf{w} > +b$, $\mathbf{w} \in \mathbb{R}^n$ and $b \in \mathbb{R}$, that can discriminate between positive $(+1)$ and negative (-1) labeled inputs. The linear decision function $f(\mathbf{x})$ is learned by solving a dual quadratic optimization problem, where $\mathbf{w}$ and b are optimized such that the margin of separation between the two classes is maximized [13].

For many problems, however, a good linear decision function $f(\mathbf{x})$ in the n-dimensional input space does not exist. In such cases, the labeled inputs can be mapped into a (possibly) higher-dimensional feature space, e.g., $\mathbf{x}_i \rightarrow \Phi(\mathbf{x}_i)$, where a good linear decision function can be found. The mapping is defined implicitly through the use a kernel function $K(\mathbf{x}_i, \mathbf{x}_j) =< \Phi(\mathbf{x}_i), \Phi(\mathbf{x}_j) >$ that is subject to Mercer's Condition [13]. The kernel function can also be considered as a measure of similarity between two input data points. When using a kernel function, the output of the function between two instances (i.e., $K(\mathbf{x}_i, \mathbf{x}_j)$) replaces their dot product in the dual quadratic optimization framework (see [13, 14] for details). Hence, the actual higher dimensional representation $\Phi(\mathbf{x}_i)$ need not be computed explicitly.

In the experiments conducted, the polynomial kernel function with exponent 2.0 was used [14]. The pairwise-coupling method of Hastie *et al.* [15] was applied to generalize the original binary classification SVM algorithm to the multi-class problem of object recognition. The SVM implementation in the WEKA machine learning library [12] was used, which implements the sequential minimal optimization algorithm for training the model [16]. To obtain a probabilistic estimate for the class label of a test data point, logistic regression models are fit to the outputs of the SVM, as described in [12].

3) Bayesian Networks: Bayesian network is a generative probabilistic graphical model that represents a set of variables and their probabilistic independencies [17]. Formally, a Bayesian network $< G, \Theta >$ is defined over a set of variables $\mathbf{x} = x_1, \ldots, x_n$, such that G is a directed acyclic graph whose nodes represent the variables $\mathbf{x}$, and $\Theta = \{\theta_i\}$ represent the set of parameters defining the conditional probability of each node in the graph given its parents, i.e., $Pr(x_i|x_{par(x_i)}, \theta_i)$ where $x_{par(x_i)}$ is the set of parent nodes of x_i in G. Due to space constraints, the reader is referred to [17] for details regarding the Bayesian network model.

A Bayesian network can be learned from a set of data by inducing the network structure G and estimating the parameters Θ that maximize some particular objective function (e.g., log-likelihood). In this study, the WEKA [12] implementation for a Bayesian network was used, which learns the network using the hill climbing algorithm proposed by Cooper *et al.* [18]. All training and inference parameters available in the WEKA implementation were set to their default values. Because Bayesian networks are designed to work on discrete data, the numeric features of each sound are discretized using the discretization filter in the WEKA library [12].

C. Performance Evaluation

The generalization performance of the models M_{grasp}, M_{push}, and M_{drop} is estimated using leave one out cross-validation. Let $\{S_i, O_i\}$, where $i = 1, \ldots, N$, be a set of data for some given behavior $B \in \mathcal{B}$. During each iteration of the cross-validation procedure, one data point from the set is used for testing and the rest $N-1$ data points are used for training the model M_B. For each behavior, there are six data points for each of the eighteen objects, resulting in $N = 6 \times 18 = 108$.

The performance of the models is reported in terms of the percentage of correct predictions, i.e., accuracy, where:

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

To evaluate whether multiple different interactions with the same object improve prediction, the predictions of the models M_{grasp}, M_{push}, and M_{drop} are aggregated using equal weight as follows. Let S_i^{grasp}, S_i^{push}, and S_i^{drop} be the sounds generated when applying the three behaviors to the same object O_i during the i^{th} trial such that neither of the three sounds appears in the training sets for the three different models M_{grasp}, M_{push}, and M_{drop}. Once the models for each behavior $B \in \mathcal{B}$ are trained, the combined prediction is then assigned to the object class $o \in \mathcal{O}$ that maximizes:

$$\sum_{B \in \mathcal{B}} Pr_B(O_i = o|S_i^B)$$

The goal of this procedure is to determine whether applying multiple distinct behaviors to the same object will result in better prediction performance. The next section summarizes the results and compares the performance of the three different learning algorithms used in the experiments.

TABLE I

ACCURACY OF THE MODELS M_{grasp}, M_{push}, M_{drop}, ALONG WITH THE COMBINED MODEL WHEN USING K-NN, SVM, AND BAYESIAN NETWORK

Behavior Model	k-NN	SVM	Bayesian Network
M_{grasp}	60.19%	77.78%	90.74%
M_{push}	72.22%	71.31%	84.26%
M_{drop}	50.93%	61.11%	60.19%
$M_{combined}$	75.00%	87.96%	97.22%

V. RESULTS

Table I shows the performance rates of the models M_{grasp}, M_{push}, M_{drop}, along with the combined model when using the three different learning algorithms: k-NN, Support Vector Machine (SVM), and Bayesian Networks. As a point of reference, a random predictor would achieve about $(1/18) \times 100\% = 5.6\%$ accuracy, given that $|\mathcal{O}| = 18$.

A. Comparison of Learning Algorithms

The first observation is that all three learning algorithms can recognize the objects in the test set significantly better than chance. Overall, the Bayesian Network learning model significantly outperforms k-NN and SVM, resulting in high accuracy rates, with the exception of the M_{drop} model whose performance is similar with both the SVM and Bayesian Network learning models.

A possible reason why the memory-based k-NN model does not perform well is that there are irrelevant features in the sound feature vector. This is likely due to the background noise produced by the ventilation and air conditioning systems in the lab. The discriminative SVM model also suffers from this drawback when using standard kernel functions (such as the polynomial or Radial Basis Function kernels). In addition, empirical results suggest that generative models (such as Bayesian Networks) achieve their asymptotic error rates with less data then their counterpart discriminative model, thus making them preferable when training data points are scarce [19]. Given that for each behavior there are only 6 data points per object class, the amount of data recorded is relatively small which may be another reason why SVM was not able to outperform the Bayesian Network model.

B. Single vs. Multiple Behaviors

The results in Table I show that object recognition based on auditory information is most difficult when using the dropping behavior, i.e., many objects sound very similar (given the feature representation used) when dropped. The grasping behavior, on the other hand, produces sound feature vectors that are most informative of the object being grasped (when using SVM and Bayesian Network).

In the case of the $push$ behavior, the Bayesian Network model makes several prediction mistakes. For example, the *rubber ball* gets mis-classified as a *tennis ball* in three out of six trials, while the *tennis ball* gets mis-classified as being a *plastic ball* in all 6 trials. The *soft plastic cup* is also mis-classified as a *hard plastic cup* in two out of four trials, while the *pop can* gets classified as a *plastic box* once. These mistakes show that given the feature representation, there are pairs of objects that sound very similar when a given behavior is applied to them. This was true for all three behaviors.

The Bayesian Network combined model, $M_{combined}$, makes only three prediction errors: the *whiteboard eraser* is mis-classified as a *hockey puck* once, the *tennis ball* is mis-classified as a *rubber ball* once; and the *rubber ball* is predicted as being the *tennis ball* in one out of the six trials. The low error rate shows that using multiple different interactions (i.e., different exploratory behaviors) with the object significantly boosts prediction performance for all three learning algorithms. This indicates that the errors that the three models, M_{grasp}, M_{push}, and M_{drop} make are uncorrelated. As noted earlier, it is very difficult to discriminate between the *tennis ball* and the *rubber ball* based on the sounds they make when applying the $push$ behavior. However, the M_{grasp} model achieves perfect classification for these two objects, thus helping resolve the ambiguity between them. Similarly, the M_{drop} model cannot distinguish well between the *metal plate* and *metal flange* objects, while the M_{push} model achieves perfect classification for these two objects.

C. Performance vs. Amount of Training Data

The Bayesian Network models (i.e., the models that achieve the best performance for this task) were also evaluated by varying the amount of data available in the training set. Figure 5 shows the accuracy rates as the number of trials with each object is varied from two to six. Even with just two trials per object (i.e., one in the training set and one in the testing set), the models M_{grasp}, and M_{push} can predict the object class significantly better than chance. As expected, access to more training data leads to improved accuracy for all models.

Finally, the performance of the three different learning algorithms was also evaluated as a function of that amount of training data that was used. Figure 6 shows the performance of the k-NN, SVM, and Bayesian Network algorithms for the

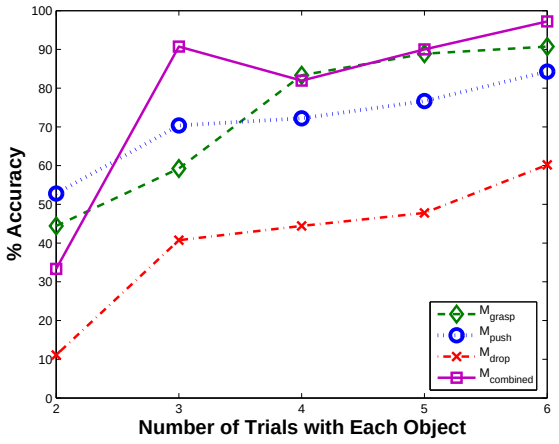


Fig. 5. Performance of the models M_{grasp}, M_{push}, M_{drop}, and $M_{combined}$ using Bayesian Network as the number of trials with each object is ranged from 2 to 6.

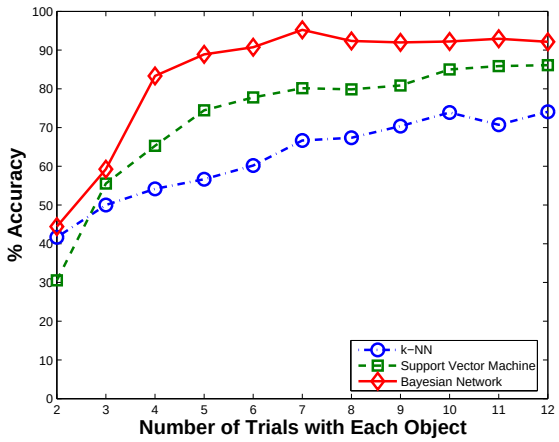


Fig. 6. Performance of the models M_{grasp} using the three different learning algorithms as the number of trials with each object is varied from 2 to 12.

model M_{grasp} as the number of trials per object is varied from 2 to 12 (six additional grasping trials were recorded for each object during the same recording session). The Bayesian Network model reaches its asymptotic error rate much quicker than the k-NN and SVM learning algorithms.

VI. Conclusions and Future Work

The study presented in this paper investigated how a robot can use auditory information to recognize the object it is interacting with. The proposed framework uses machine learning methods and different exploratory behaviors which allows the robot to predict the object class (one of eighteen possible objects) given the detected sound and behavior.

The framework presented here uses standard machine learning algorithms to solve the task. Three such algorithms were evaluated: k-Nearest Neighbor, Support Vector Machine, and Bayesian Network. The algorithms represent three major families of machine learning models: *instance-based*, *discriminative*, and *probabilistic generative graphical models*. While all models performed significantly better than chance, the Bayesian Network model was able to achieve the highest level of accuracy with the least amount of training data. Unlike the standard SVM and k-NN models, the Bayesian Network model is able to detect the irrelevant features in the sound feature vectors produced by background noise (e.g., due to the air conditioning system in the lab).

The robot used three different exploratory behaviors to interact with the objects: grasping, pushing, and dropping. For each of the three behaviors, there were pairs of objects which sounded very similar (in terms of the feature representation that was used), e.g., the *tennis ball* and the *rubber ball* were almost indistinguishable when being pushed to the side. However, by applying multiple behaviors to the same object, the predictions of the models for each behavior can be combined which resulted in higher accuracy rates than for any single behavior performed alone. These results were consistent for all three learning algorithms evaluated in this study.

While the number of objects used in this study was relatively high (in comparison to other related work in the field, see Section II), it is still a challenging task to classify the hundreds of different objects found in human environments based solely on their acoustic properties. Different initial configurations between the object and the robot add additional difficulties to the problem. One possible approach to be investigated in future work is to use unsupervised methods (e.g., hierarchical clustering) in order to learn categories of object sounds. Such an approach would allow the robot to use standard machine learning methods for each object category (which would contain a smaller set of objects than the total set to which the robot is exposed) while at the same time learn how the categories relate to each other. Also, more powerful feature representations may be used in order to capture and model the temporal patterns in some objects interactions (e.g., periodicity of bouncing).

References

[1] W. W. Gaver, "What in the world do we hear? An ecological approach to auditory event perception," *Ecological Psychology*, 1993.
[2] M. Grassi, "Do we hear size or sound? Balls dropped on plates," *Perception and Psychophysics*, 2005.
[3] C. A. Fowler, "Auditory perception is not special: We see the world, we feel the world, we hear the world," *Journal of Acoustical Soceity of America*, 1991.
[4] E. Krotkov, R. Klatzky, and N. Zumel, "Robotic perception of material: Experiments with shape-invariant acoustic measures of material type," in *Experimental Robotics IV*, ser. Lecture Notes in Control and Information Sciences. Springer Berlin/Heidelberg, 1996, vol. 223, pp. 204–211.
[5] R. L. Klatzky, D. K. Pai, and E. P. Krotkov, "Perception of material from contact sounds," *Presence: Teleoperators & Virtual Environments*, vol. 9, no. 4, pp. 339–410, 2000.
[6] J. L. Richmond and D. K. Pai, "Active measurement of contact sounds," in *Proc. of the IEEE Conference on Robotics and Automation*, 2000.
[7] J. L. Richmond, "Automatic measurement and modelling of contact sounds," Master's thesis, University of British Columbia, 2000.
[8] E. Torres-Jara, L. Natale, and P. Fitzpatrick, "Tapping into touch," in *Proc. of the Fifth International Workshop on Epigenetic Robotics*, 2005.
[9] K. E. Lee, H. W. Hon, and R. Reddy, "An overview of the SPHINX speech recognition system," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 38, no. 1, pp. 35–45, 1990.
[10] W. D. Aha, D. Kibler, and M. K. Albert, "Instance-based learning algorithm," *Machine Learning*, vol. 6, pp. 37–66, 1991.
[11] C. G. Atkeson, A. W. Moore, and S. Schaal, "Locally weighted learning," *Artificial Intelligence Review*, vol. 11, no. 1-5, pp. 11–73, 1997.
[12] I. H. Witten and E. Frank, *Data Mining: Practical machine learning tools and techniques*, 2nd ed. San Francisco: Morgan Kaufman, 2005.
[13] V. Vapnik, *Statistical Learning Theory*. New York: Springer-Verlag, 1998.
[14] C. J. Burges, "A tutorial on support vector machines for pattern recognition," *Data Mining and Knowledge Discovery*, vol. 2, pp. 121–167, 1998.
[15] T. Hastie and R. Tibshirani, "Classification by pairwise couplling," *Advances in Neural Information Processing Systems*, vol. 10, 1998.
[16] S. Keerthi, S. Shevade, C. Bhattacharyya, and K. Murthy, "Improvements to Platt's SMO algorithm for SVM classifier design," *Neural Computation*, vol. 13, no. 3, pp. 637–649, 2001.
[17] J. Pearl, *Probabilistic Reasoning in Intelligent Systems*. Morgan Kaufmann, 1988.
[18] G. Cooper and E. Herskovits, "A bayesian method for the induction of probabilistic networks from data," *Machine Learning*, vol. 9, no. 4, pp. 309–347, 1992.
[19] A. N. Ng and M. Jordan, "On discriminative vs. generative classifiers: A comparison of logistic regression and naive bayes," *Advances in Neural Information Processing Systems*, vol. 14, 2002.