Learning to Detect Doorbell Buttons and Broken Ones on Portable Device by Haptic Exploration In An Unsupervised Way and Real-time

Liping Wu

April 21, 2011

Abstract

The paper proposes a framework so that the robot can learn to detect the doorbell buttons on a portable device and identify the broken ones by haptic exploration and in an unsupervised way. 4 doorbell buttons and 1 wood were mounted onto a portable device made for the robot to generate button surfaces and non-button surfaces. The portable device was made for the robot on purpose so that it can match the robot's hand very well and can be grasped stably by two fingers of the robot hand while the robot hand frees the third finger to press the surface. The empirically optimal exploratory procedure, press exploratory procedure, is chosen to obtain the haptic property of doorbell buttons. The press procedure is generated by closing/opening the free finger, which is straightforward selection considering the operation with the portable device.

Joint torque sensor in the distal joint of the free finger is used to guide the close/open of the finger as well as notify the touch of surface. Motor position sensor is used to measure the travel distance during touch and the microphone sensor is used to get the maximal volume heard during touch. The unsupervised learning algorithm, *k*-means is used to do the learning work and is run in two clustering procedures. The first clustering procedure considers the travel distance only and partitions the exploration trails into *button-press* or *non-button-press*. The second clustering procedure considers the maximal volume only and partitions the exploration trials into *bell-triggering-press* or *non-bell-triggering-press*. Buttons are detected if the trial is clustered into *button-press* after the first clustering procedure. Broken buttons are identified if the trial is clustered into both *button-press* and *non-bell-triggering-press* after these two clustering procedures. Results show that the robot can learn to consistently detect buttons and broken ones.

Real-time detection and on-line learning are achieved by implementing the framework in C/C++ program. The control of the closing/opening of the finger, the reading of

joint torque and motor position, and the extraction of audio stream (OpenAL) are combined into the C/C++ program and made parallel using pthread. The two clustering procedures are also implemented in C/C++ functions and called to get the detection result right after the completion of a press trial. The data associated with the trial explored is updated into the experience data base right after the completion of the trial so that the robot can learn on-online.

1. Introduction

The study focuses on detecting the doorbell buttons and broken ones on a portable device, which made by the author particularly to match the robot's hand. Portable devices with buttons on are very common in human's life. For example, TV remote controller with varied buttons is used to choose the channels and adjust the volume of TV. People hold the flashlight to light up and press the button on the flashlight to turn on/off the light. Buttons are everywhere in human's life and doorbell buttons are representative among kinds of buttons. Therefore, this study is meaningful in building the robot in helping human's life.

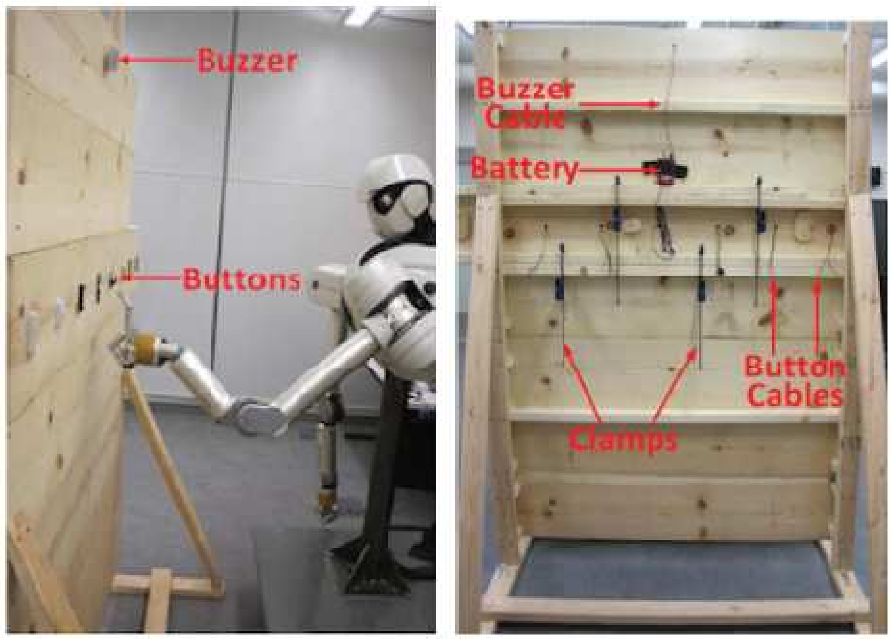
Previous work in detecting buttons shows interest in using vision (Miura et. al 2005, Klingbeil et. al 2008, Sukhoy et. al. 2010). Their work aim to let the robot learn and get the visual representation of the button so that the model learned can be used to detect the button in visual space. However, rare work is done to learn and get the haptic representation of the button, so that the robot can find the buttons by only haptic exploration and modalities in the absence of vision. It should also be very important to get the haptic representation of the button. Not only the robot can still detect and locate and operate the buttons under the condition where there is no vision available, but also the haptic representation is more accurate in defining a button, which is invented by human to help human's life.

More correctly, this research focuses on the push-button, which has a spring in to return to the un-pushed state. Because of the spring in the push-button, the fingertip can sink in while keep contact with the button's surface and feel the resistance, and also when being released the button's surface will resume automatically. (Sukhoy and Stoytchev 2010) trained a visual model that can detect the button with the button-like texture in vision. However, any object with the button-like texture but without the spring and the tactile and proprioceptive properties above, we still can't say that it is a push-button. Therefore, the representation for a button derived from its tactile property is more accurate and may be able to achieve a more accurate detection result.

From the point of view of haptic exploration, sensing the push-button is also very meaningful. By haptic exploration, humans can learn many characteristics of objects, such as object shape, surface texture, stiffness and temperature. This kind of research is also viewed as the tactile data interpretation, which supports the dexterous manipulation a lot. For example, (Okamura and Cutkosky 1999a) designed a mathematical model based on a differential geometry approach to detect small surface feature of bump.

2. The Previous Study

This project is based on the previous study by the research group in Developmental Robotics Laboratory at Iowa State University. The robot for the previous study as well as this project proposed is showed in Figure 1. This robot has two Barrett Whole Arm Manipulators (WAMs) with a BH8-Series Barrett Hand as arms. Two Logitech cameras are mounted in its head as his eyes. It also has a microphone mounted on the head and an artificial fingernail attached on the finger 3 on the left arm.



(a) The robot pushing a button (b) Experimental fixture (back)

Figure 1: The robot and fixture for the experiment.

In the previous study, there are mainly two projects. One project (Sukhoy, Sinapov, Wu and Stoytchev 2010) is humanoid robot learning to press doorbell buttons using active exploration and audio feedback. With 5 sampled points in the 7-D joint space, the robot can calculate itself to generate press behaviors of pressing an area on the board. The press behaviors are parameterized by the vector decided by the start position and end position of the behavior in the 7-D joint space. By running the pre-learned classifier on the audio stream, the time when the doorbell is triggered can be detected in real-time. For each behavior, it will be labeled as pressing a button or not pressing a button according to if there is a doorbell detected at the meantime. Finally, k-nearest neighbor algorithm is used to do the learning work and three kinds of active selection strategies—random exploration, uncertainty-driven exploration, and stimulus-driven exploration—are used to speed up the learning.

In another project (Sukhoy and Stoytchev 2010), the humanoid robot learns the visual model of doorbell buttons autonomously. Color tracker is used to track the touch position on the board surface in the image from robot’s camera. These touch position is simplified as a pixel in the image and labeled as the functional component or not according to if the associated press behavior is pressing button or not based on the audio feedback. Image is split into 10x10 pixel patches, and each patch is labeled as functional component or not according to the density of functional component touch point falling into the grid. For each patch, the texture, edge and low-frequency color information of itself and neighbors are extracted and logistic regression classifier is used to learn the visual model for detecting patches belonging to the functional component of the doorbell buttons.

3. Related Work

3.1 Button Study

In psychology, (Hauf and Aschersleben 2008) found that a 9-month old infant can anticipate what color of buttons will trigger the light or the ring when he/she presses from experience, and in turn by the anticipation control his/her action to press the working buttons more often. In the experiment, the infants were placed in front of 3 groups of buttons. In the first group, the red button is effective. In the second group, the blue button is effective, and in the third group, none button is effective. The result shows that the infants press red button more often for the first group, blue button more often for the second group and almost the same for the third group.

In robotics, the previous work focuses on the visual feedback more. (Thomaz 2006) used social learning to teach the robot how to turn the button on & off using speech communication. But the robot uses the vision to recognize where the button is and decide if the button is on or off. (Miura, Iwase, and Shirai 2005) made the robot execute a take-an-elevator task based on vision. Vision-based teaching algorithm was used to find the location of the elevator door and elevator button. The origin of the elevator was marked with a red light, and the robot searched the area around the origin to find the image template of the elevator. Similarly, being indicated the rough position of the buttons; the robot finds the position of the button by searching for the area nearby. (Klingbeil, Saxena and Ng 2008) tried a haar classifier using supervised learning algorithm for the robot to detect where the elevator button is.

3. 2 Haptic Exploration

In psychology, haptic exploration is defined as exploratory procedures (EPs) related with the modality of touch. EPs are stereotyped patterns describing the ways of contact and movement between skin and object (Lederman and Klatzky 1987). During exploration, the perceptual system, haptics, incorporates inputs from multiple sensory systems (Loomis and Lederman 1986). Haptics includes a cutaneous system sensing pressure, vibration, temperature, and a kinesthetic system registering position and movement of the muscles and joints. Between EPs and object properties, there are associations describing whether an EP is necessary, optimal, sufficient, or inadequate in exposing a specific property of an object (Klatzky, Lederman, and Matula 1991). By haptic exploration, human can learn these associations, which, in turn, can help the human to choose an optimal EP for obtaining the desired object property. Empirically, the press EP is optimal in obtaining the press feeling of a push-button. For human, a press EP means using the fingertip to add external force perpendicularly onto the surface. In this study, the press EP is generated by closing one finger onto the surface, which is held stably in the robot hand's palm.

For the studies of haptic exploration in robotics, most of them focus on detecting the object shape (Caselli et al. 1996, Allen and Roberts 1989, Roberts 1990) and small surface features such as cracks, bumps and ridges (Okamura et al. 2000, Okamura and Cutkosky 2001, Okamura and Cutkosky 1999b). Some papers also designed the models to measure surface toughness, friction, and texture (Okamura et al. 2000, Stansfield 1992, Sukhoy et al. 2009). (Stansfield 1992) used two kinds of pressure

EPs to measure the hardness of objects. One is by grasping and squeezing the object, and another one is by probing against the object surface using one finger. In our study, the later kind of EP will be used to detect the push-button, which, to some degree, can be viewed as a soft object that can be probed in.

4. Haptic Feeling for a Press on Doorbell Button

For human, a press EP means using the fingertip to add external force perpendicularly onto the surface. In this study, the press EP is generated by closing one finger and putting fingertip onto the surface, which is held stably in the robot hand's palm. The surface is held in the palm in such way that it is parallel to the palm surface. The rotation direction of the finger is perpendicular to the palm surface, and also the fingertip is spherical. Therefore, the external force added onto the target surface will be perpendicular within the range of acceptable error when close the finger onto the surface.

The doorbell button in this study is a push-to-make push button. A push-button (also spelled pushbutton) or simply button is a simple switch mechanism for controlling some aspect of a machine or a process. Most of the buttons are biased switches. There are two types of biased switch, and they are push-to-make and push-to-break. For a push-to-make button, contact is made when pressed and broken when released. On the contrary, contact is broken when released and made when pressed for a push-to-break button.

Most of buttons are push-to-make type, such as computer keyboard and doorbell button, which is the research target of this paper. The function for a push-to-make type button is to make contact by narrowing the distance due to the loading of the external force, and to break contact by broadening the distance due to the unloading of the external force. Therefore, the correct haptic feeling when doing a press on a push-button is, there is considerable displacement change along the force change direction.

Because of the spring in the button, there are also the associated characteristics, say buffing effect. This effect may be observed from the collision of fingertip and surface during the short time when the fingertip hits the surface. The vibration in the interaction force will be more smoothly for a button than for a hard non-button surface. However, these are minor comparing the travel distance property resulting from the function of a button, and are ignored in this study.

5. Experimental Setup

5.1 Robot Hand

The robot has two BH8-Series Barrett Hands (Figure 2). The BH8-Series Barrett Hand has three fingers. Each of them can be controlled independently to close completely, open completely, close the given number of counts, or open the given number of counts. In the distal joint of each finger (Figure 3), there is a strain gage joint-torque sensor, which can measure the force applied to the finger tip, Force A. Moreover, the two joints in the finger are controlled by only one motor, so that the position of the whole finger can just be decided by the position of the motor associated to the finger.

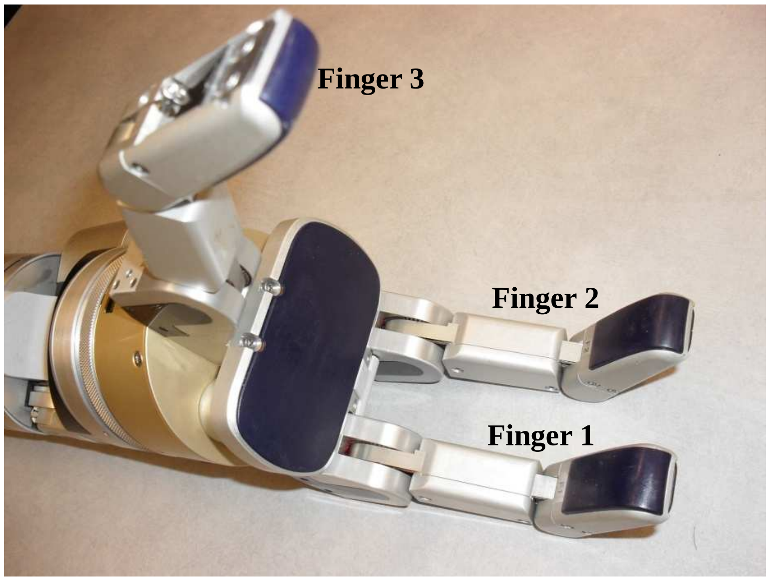


Figure 2. 3-finger Barrett robot hand used in the study

Both the strain gage joint-torque and the motor position will be read in 50 Hz. Although these two sensors can be read in a much higher frequency, 50 Hz will be enough since the finger will be controlled to close in real time in a low frequency of 10 HZ. Every 100 milliseconds, the motor will be input a command of closing a small number of counts and can be stop at specific case. There is also a microphone mounted into the robot's head, and it is read in 44.KHz.

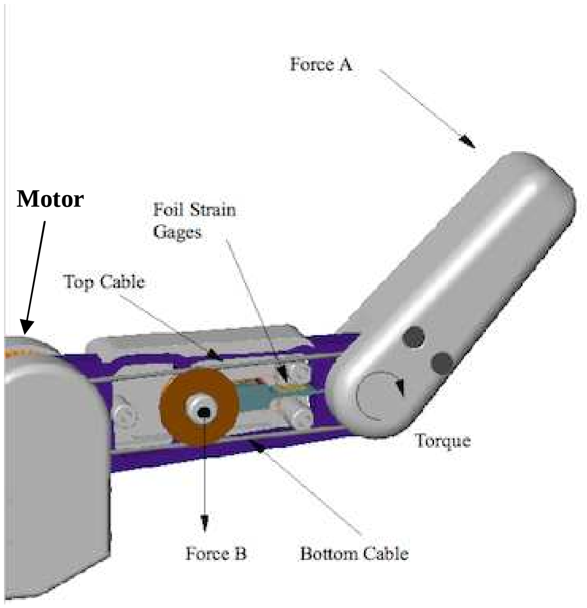


Figure 3. Strain Gage Joint-Torque Sensor and Motor

5.2 Portable Device

In the experiment, the robot will grasp a portable device in hand and it will close/open one finger iteratively to press one part of the surface on the device. The goal of this project is offering the robot such ability that it can learn to distinguish pressing a button from pressing a non-button on a portable device, and distinguish *bell-triggering-button-pressing* from *non-bell-triggering-button-pressing* on a portable device. The button will be the common doorbell button in human life, since the robot is expected the potential meaning in making human's life easier.

Human designs kinds of portable devices for themselves so that they can hold these devices more stably and more comfortably. The robot hand in this study has only three hands and is also much different from human's hand in shape and material. To make the robot can use the portable devices designed for human hand, the straightforward solution is just creating a more human-like hand for the robot, which is obvious not the work in this project. Therefore, a unique portable device was made by the author for matching the robot hand so that the robot can grasp the device using two fingers stably while free the third finger to do the press iteratively.

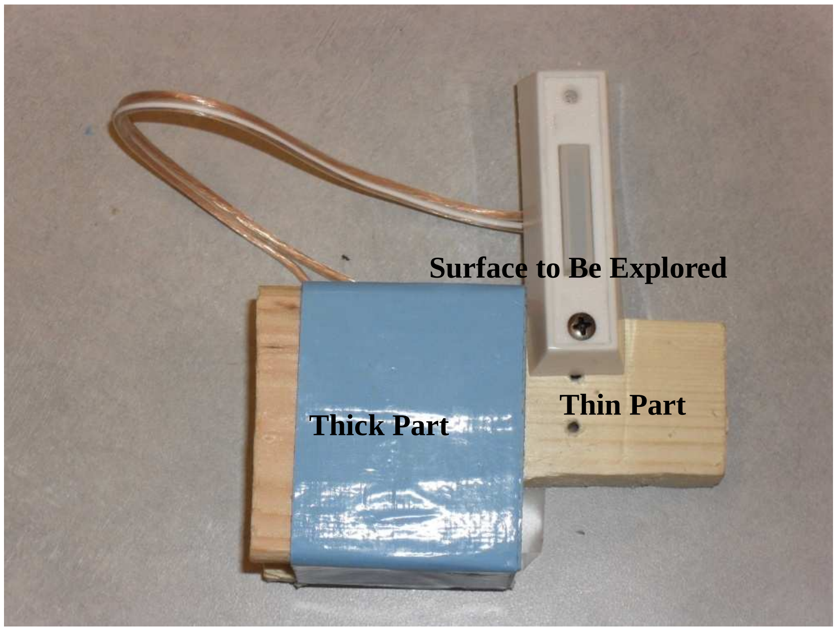


Figure 4. Portable device made for the robot

Figure 4 shows you the portable device made for the robot. The robot uses finger 1 and finger 3 to grasp the thick end of the device and the doorbell button is mounted to the thin end of the device so that the robot can close the finger 3 deep enough to press the button (Figure 5).

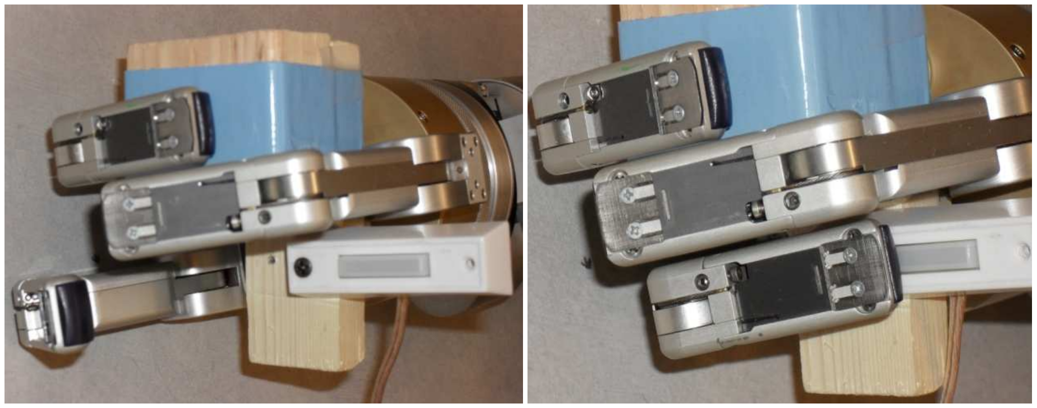

(a) (b)

Figure 5. Portable device being grasped by the robot hand: (a) with finger 2 open; (b) with finger 2 closed.

5.3 Doorbell Buttons and Non-Button-Surfaces and Bell

Figure 6(a) shows you the four buttons and one extension wood used in this study. For getting the button-surfaces (Figure 6(b): surfaces #1, 3, 5, 7.), the buttons were mounted onto the portable device in such way that the finger 2 will press onto the moveable part of the buttons. To get the non-button surfaces (Figure 6(c): surfaces #2, 4, 6, 8.) as well as eliminate the impact of the device, the buttons will be mounted with a few offsets from the ways for getting button-surface so that the finger 2 will press onto the unmovable edge of the button instead. Moreover, a wood will be mounted to extend the thin part of the portable device to get another non-button-surface (Figure 6(c): surface # 9). Figure 6(c) shows you all the non-button-surfaces used in this study.

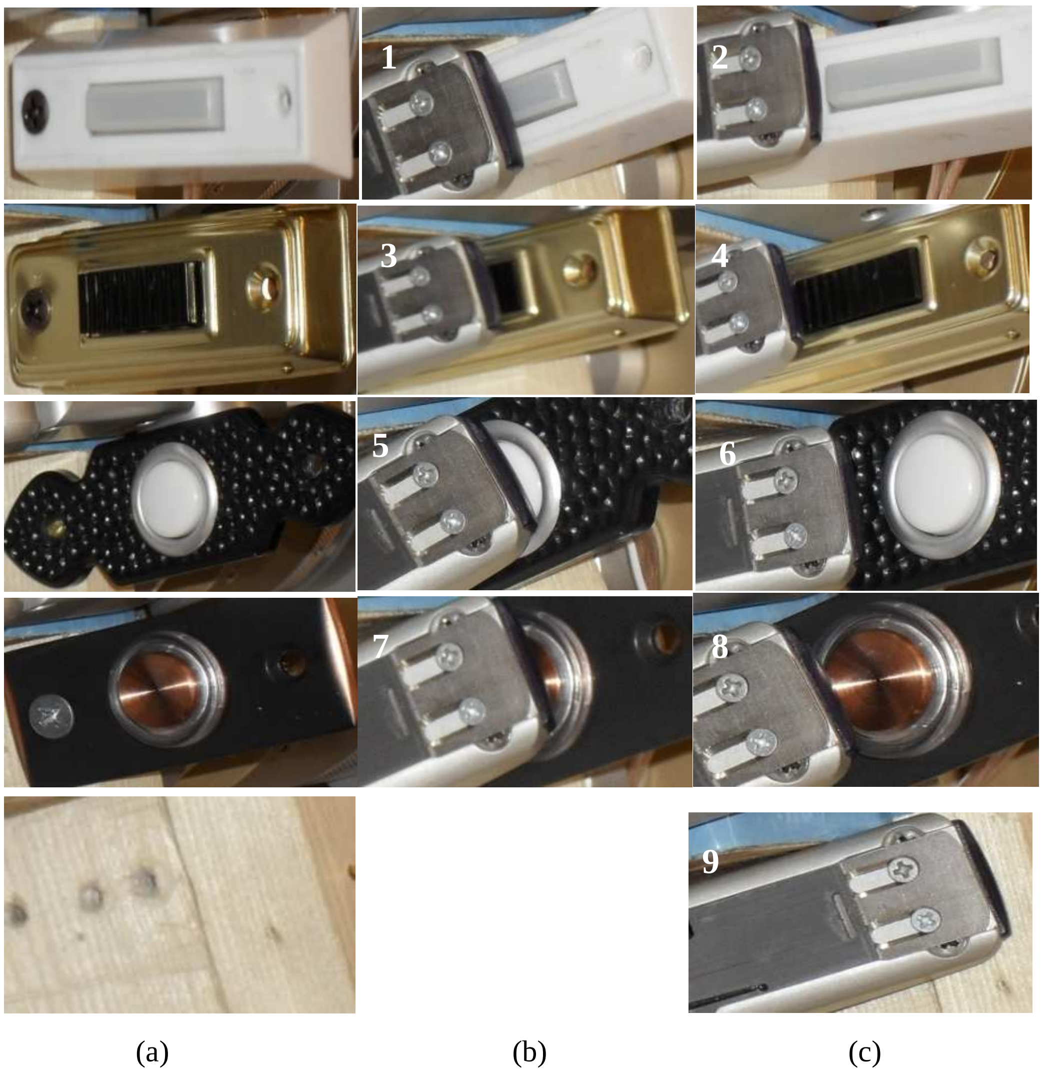


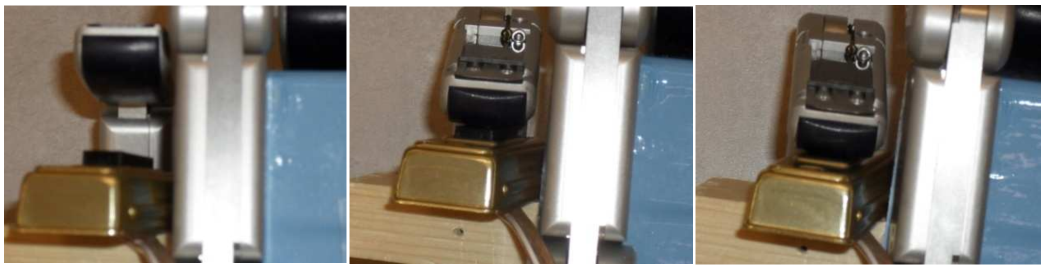
Figure 6. Surfaces explored by the robot: (a) 4 buttons and 1 wood generating the surfaces; (b) 4 button surfaces generated from the 4 buttons and with fingertip right above the moveable part; (c) 4 non-button surfaces generating from the 4 buttons and with fingertip on the non-moveable edge, and 1 non-button surface made from extension wood.

So, 4 button surfaces and 5 non-button surfaces (9 surfaces in total) will be explored in this study. A door bell mounted onto a board which stands perpendicularly in front of the robot will be connected to the buttons when necessary to generate bell-triggering-button-pressing. The setup for door bell is showed in Figure 1 in the previous study.

5.4 Press Exploratory Procedure

The press Exploratory Procedure (EP) will be used to obtain the haptic property of the button. The press EP is generated by closing the finger 2 onto the surface. The closing of the finger 2 will consist of two steps. During the first step, in a frequency of 10 Hz, the motor associated with finger 2 will execute a command for closing a certain number of counts when the notification state remains non-touching, which means the finger is not touching a surface. At the mean time, the joint torque in the distal joint of the finger will be read in a frequency of 50 Hz and checked if it exceeds a specific threshold. If it does, then the notification state will change to touching from non-touching and notify the robot that the finger now is touching a surface and the closing in first step should be stopped (Figure 7(b)).

The second step of closing starts when the surface is being touched. Finger 2 will now execute a closing command which closes the finger until the torque in the motor reaches a specific limit or until the motor position reaches the destination (Figure 7(c)). In this study, this step of closing will be designed on purpose so that it is the torque limit of motor but not the destination of the motor position is reached. In this case, different surfaces will receive the same press force within the range of acceptable error at the end of the second step of closing. Figure 7 shows you the position of the fingertip of finger2 during the press exploratory procedure on surface #1 (button surface).



(a) (b) (c)

Figure 7. Position of the fingertip of finger2 during the press exploratory procedure on surface #1 (button surface): (a) ready for press; (b) the end of first step as well as the start of the second step; (c) the end of the second step.

6. Methodology

6.1. Data Collection

After one press EP, the finger 2 will be reset in order to be ready for another press EP. It will be open to a specific position where the finger tip of the finger will be away from the surface totally and the joint torque in the distal joint will be below the threshold notifying touching. One trail consists of the press EP and the open for reset. For each trial, both the travel distance of the motor position associated to the finger 2 and the maximal volume heard from the microphone during the second step of closing will be recorded in real-time for further analysis. For each non-button surface, 6 trials will be performed. For each button surface, 3 trials will be performed when it is connected to the door bell, and another 3 trials will be performed when it is not connected to the door bell. Therefore, there are 6 trials for each surface, and there are $6 \times 9 = 54$ trials in total.

6. 2. The *k*-means Clustering

The k-means Clustering algorithm (MacQueen 1967) will be used to do the learning work in this study. It is a method of cluster analysis in statistics and data mining as well as machine learning. It partitions n observations into k clusters so that each observation will belong to the cluster, which has the nearest mean to it.

Assume there are n observations $(x_1, x_2, \dots, x_n)$ and each observation is a d-dimensional real vector. These n observations will be clustered into k sets $(k \le n)$ $\boldsymbol{C} = \{C_1, C_2, \dots, C_k\}$. Then, k-means clustering aims to do the clustering so as to minimize the within-cluster sum of squares (WCSS):

$$\underset{\mathbf{C}}{\operatorname{argmin}} \sum_{i=1}^{k} \sum_{x_j \in C_i} ||x_j - \mu_i||^2$$

where μ_i is the mean of observations in C_i.

6.3. Normalization before Clustering

To reduce the effect of unit and scalability, the parameters of each observation need to be normalized into the range [0, 1] before run clustering algorithm on the observations set. Assume observation vector x_j $(1 \le j \le n)$ consists of p parameters and we have $x_j = \left(p_1, p_2, \dots, p_p\right)$ $(1 \le j \le n)$.

Get the new observation vector $\widehat{x_j} = \left(\widehat{p_1}, \widehat{p_2}, \dots, \widehat{p_p}\right)$ $(1 \le j \le n)$ by following normalization formula:

$$\widehat{p_l} = \frac{p_l - p_{min}}{p_{max} - p_{min}} \, (1 \le l \le p)$$

where $p_{min} = min_{1 \le l \le p}\, p_l$ and $p_{max} = max_{1 \le l \le p}\, p_l$.

Then, new observation vectors will be fed into the clustering procedure instead.

6. 4. Detecting Button and Identifying Broken Button

Take these 42 trials as 42 observations. For solving these two tasks, two k-means clustering procedures will be run on these observations to get two clusters for each observation. For the first clustering procedure, each observation will only have travel distance as its parameter, and it will be clustered into two clusters, *button-press* or *non-button-press*. For the second clustering procedure, each observation will only have maximal volume as its parameter and it will be clustered into two clusters, *bell-triggering-press* or *non-bell-triggering-press*.

Therefore, the task of detecting button can be solved using the first clustering procedure. As long as a press is identified as *button-press* after the first clustering procedure, the robot can know that it is pressing a button so that it can detect the button. For solving the task of identifying broken button, both of these two procedures need to be run. If a press is identified as *button-press* as well as *non-bell-triggering-press*, the robot will know it is pressing a broken button, which may be disconnect to the bell and can't function correctly to trigger the bell.

7. Results

7.1. Detecting Button and Identifying Broken Button

Table 1 shows some sample results for the exploratory trials on surface1 and surface 8. When use the openAL program to extract the volume value from the microphone device, the value will be stored into a signed 16 bits integer and may vary from 0 to 32767 (although the maximal value for a signed 16 bits integer is 65535). Bigger value means higher volume. When the finger 2 is open completely, the motor position is about 10. On the contrary, when the finger 2 is closed completely, the motor position is about 200000. For the first cluster procedure, the cluster 0 turns out to be the cluster of *button-press*, but for the second cluster procedure, the cluster 0 turns out to be the cluster of *bell-triggering-press*.

Table 1. Results of exploratory trials on surface 3 and surface 4

Trial ID	Surface		Connect to Bell (Y/N)	Travel Distance		Maximal Volume		Cluster assigned	
	ID	type		Original	Normalized	Original	Normalized	First	Second
1	3	button	Y	4295	0.95608	32767	1	0	0
2	3	button	Y	4403	1	32767	1	0	0
3	3	button	Y	4343	0.9756	32767	1	0	0
4	3	button	N	4326	0.968686	3542	0.022543	0	1
5	3	button	N	4320	0.966246	3136	0.008964	0	1
6	3	button	N	4340	0.97438	3134	0.008897	0	1
43	4	non-button		2162	0.088654	2868	0	1	1
44	4	non-button		1944	0	3382	0.017191	1	1
45	4	non-button		1974	0.0122	3249	0.012743	1	1
46	4	non-button		1998	0.02196	3803	0.031272	1	1
47	4	non-button		1980	0.01464	3528	0.022074	1	1
48	4	non-button		1971	0.01098	3501	0.021171	1	1

Note: The clustering results consider only these 12 trials in the table.

From the Table 1, we can found that the travel distance value for a button surface is around 4000, and around 2000 for the non-button surface, which means the haptic sensor works very well in the task. Moreover, the maximal volume is around 30000 when a bell is heard, and around 3000 when no bell is heard, which means the microphone works very well too. Table shows the clustering results considering only these 12 trials in the table. After the first clustering procedure, trials are partitioned into *button-press* and *non-button-press* with a precision of 100%. After the second clustering procedure, trials are partitioned into *bell-triggering-press* and *non-bell-triggering-press* with a precision of 100%, too. So, the clustering algorithm also works very well.

The results for all the trials can be found in Table 3. Table 2 also summarizes all of the results. In the first cluster procedure, the cluster 0 is associated with the cluster of *button-press*. In the second cluster procedure, the cluster 1 turns out to be the *bell-triggering-press* cluster. For both of the cluster procedures, the percentage of incorrectly clustered trials is 0%. Therefore, we can say, with the sensors and exploratory behaviors and clustering algorithms, our robot can successfully learn to detect buttons and identify the broken buttons.

Table 2. Summary on the clustering results for all trials

	Manual clusters	Trials assigned to cluster 0	Trials assigned to cluster 1	Incorrectly clustered trials	Incorrect percentage
First cluster procedure	button-press	24	0	0	0%
	non-button-press	0	30	0	
Second cluster procedure	bell-triggering-press	0	12	0	0%
	non-bell-triggering-press	42	0	0	

Note: For the first cluster procedure, the cluster 0 is associated with the cluster of *button-press*. For the second cluster procedure, the cluster 1 turns out to be the *bell-triggering-press* cluster.

Table 3. Result table for all of the 54 trials

Trial ID	Surface		Connect to Bell (Y/N)	Travel Distance		Maximal Volume		Clustering	
	ID	Type		Original	Normalized	Original	Normalized	First	Second
1	3	button	Y	4295	0.94492	32767	1	0	1
2	3	button	Y	4403	0.981868	32767	1	0	1
3	3	button	Y	4343	0.961341	32767	1	0	1
4	3	button	N	4326	0.955525	3542	0.033916	0	0
5	3	button	N	4320	0.953472	3136	0.020495	0	0
6	3	button	N	4340	0.960315	3134	0.020429	0	0
7	9	nonbutton		1955	0.144372	3365	0.028065	1	0
8	9	nonbutton		1925	0.134109	3580	0.035172	1	0
9	9	nonbutton		1872	0.115977	3102	0.019371	1	0
10	9	nonbutton		1854	0.109819	3752	0.040858	1	0
11	9	nonbutton		1817	0.09716	3059	0.01795	1	0
12	9	nonbutton		1812	0.09545	3781	0.041817	1	0
13	1	button	N	4456	1	3269	0.024892	0	0
14	1	button	N	4104	0.879576	3695	0.038974	0	0
15	1	button	N	3825	0.784126	3434	0.030346	0	0
16	1	button	Y	4201	0.912761	32767	1	0	1
17	1	button	Y	4082	0.872049	30424	0.922548	0	1
18	1	button	Y	4114	0.882997	32148	0.979538	0	1
19	6	nonbutton		1973	0.15053	3859	0.044395	1	0
20	6	nonbutton		1533	0	4069	0.051337	1	0
21	6	nonbutton		1537	0.001368	3073	0.018413	1	0
22	6	nonbutton		1538	0.001711	2516	0	1	0
23	6	nonbutton		1546	0.004447	3084	0.018776	1	0
24	6	nonbutton		1548	0.005132	3620	0.036495	1	0
25	5	button	N	3921	0.816969	3429	0.030181	0	0
26	5	button	N	3937	0.822443	3219	0.023239	0	0
27	5	button	N	3896	0.808416	3249	0.024231	0	0
28	5	button	Y	3901	0.810127	29886	0.904763	0	1

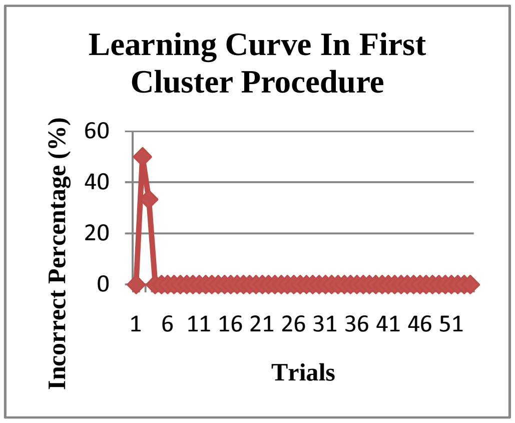
Table 3 (continued)

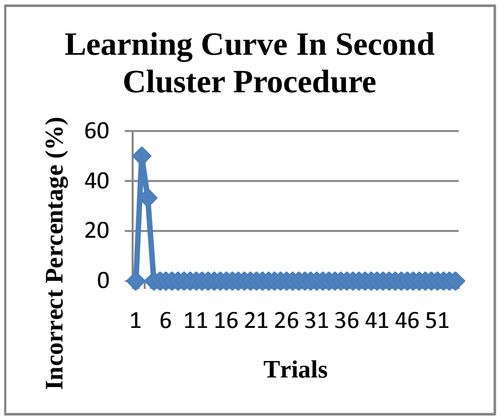
29	5	button	Y	3891	0.806705	32767	1	0	1
30	5	button	Y	3874	0.800889	32767	1	0	1
31	7	button	N	3895	0.808074	3280	0.025255	0	0
32	7	button	N	3570	0.696887	3771	0.041486	0	0
33	7	button	N	3661	0.728019	3282	0.025321	0	0
34	7	button	Y	3609	0.710229	32091	0.977654	0	1
35	7	button	Y	3557	0.692439	32767	1	0	1
36	7	button	Y	3574	0.698255	32767	1	0	1
37	8	nonbutton		2126	0.202874	3356	0.027768	1	0
38	8	nonbutton		2181	0.22169	3412	0.029619	1	0
39	8	nonbutton		2178	0.220664	3471	0.031569	1	0
40	8	nonbutton		2132	0.204926	3126	0.020165	1	0
41	8	nonbutton		2137	0.206637	3858	0.044362	1	0
42	8	nonbutton		2133	0.205269	2772	0.008463	1	0
43	4	nonbutton		2162	0.21519	2868	0.011636	1	0
44	4	nonbutton		1944	0.140609	3382	0.028627	1	0
45	4	nonbutton		1974	0.150872	3249	0.024231	1	0
46	4	nonbutton		1998	0.159083	3803	0.042544	1	0
47	4	nonbutton		1980	0.152925	3528	0.033453	1	0
48	4	nonbutton		1971	0.149846	3501	0.032561	1	0
49	2	nonbutton		2398	0.295929	3087	0.018875	1	0
50	2	nonbutton		2278	0.254875	3620	0.036495	1	0
51	2	nonbutton		2297	0.261375	3285	0.025421	1	0
52	2	nonbutton		2302	0.263086	3396	0.02909	1	0
53	2	nonbutton		2202	0.228874	3066	0.018181	1	0
54	2	nonbutton		2287	0.257954	3866	0.044627	1	0

Note: Travel Distance varies from 0 to 200000; Maximal Volume varies from 0 to 32767; cluster 0 in first cluster procedure is associated with *button-press*, and cluster 1 in the second cluster procedure turns to be the *bell-triggering-press*.

7.2. Learning Curves

Figure 8 shows you the learning curves of the robot with the number of exploration trials increases. In the first cluster procedure, the first 6 trials are in the same cluster, *button-press*. Without the samples of another cluster, the *k*-means does random selection when the number of trials is less than or equal to 3, which generates some incorrectly clustered trials. However, when the number of trials is greater than 4 or there are samples of another cluster added, the learning curve in first cluster procedure keeps at an incorrect percentage of 0% (figure (a)), which means the robot learns to detect *button-press* consistently. Similar, for the learning curve in second cluster procedure (figure (b)), the incorrect percentage vibrates away from 0% before 4 trials, but keeps at 0% consistently after the samples of another cluster is added at 4th trials. Both curves show that the robot can learn to detect buttons and identify the broken ones consistently after some trials (3 trials in our experiment) of exploration.





(a) (b)

Figure 8. Learning curves of the robot with the number of exploration trials increases: (a) in first cluster procedure; (b) in second cluster procedure.

7.3. Real-time Detection

For achieving real-time detection, the close/open of robot finger, the reading of joint torque and motor position, the extraction of audio stream and the on-line *k*-means clustering are combined in one integral C/C++ program. Figure 9 shows you the flow chart. Pthread is used to achieve the parallelization among the close/open of robot finger, the reading of joint torque and motor position and the extraction of audio stream. OpenAL is used to extract the audio stream in real-time. Whenever a trial is done, the clustering function is called twice to do the two clustering procedures and output the clustering results. Therefore, the result of detecting buttons and identifying broken buttons can be got right after the completion of the trial, which means that the robot can do the detection in real-time. The robot can also learn on-line. As soon as one trial is completed, the data for this trial will be added into the database so that the experience database will be updated on-line to ensure the robot can learn on-line.

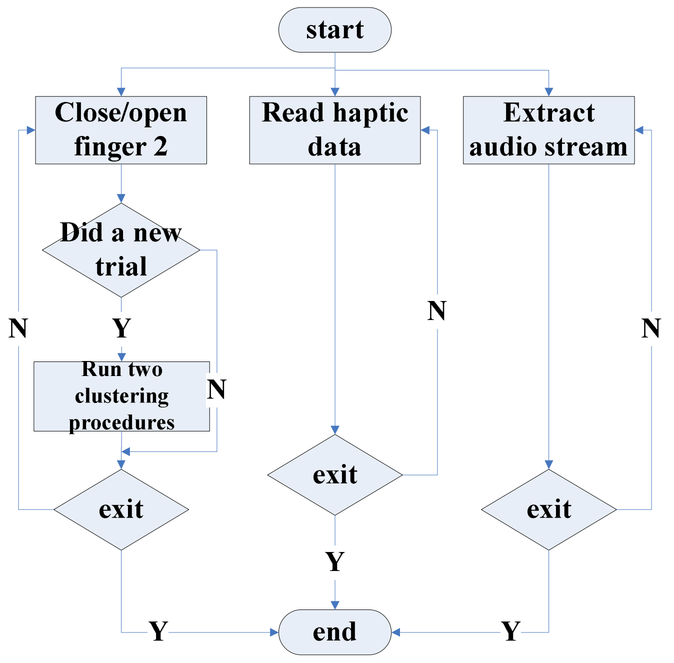


Figure 9. Flow chart of the program for real-time detection

8. Conclusion and Future Work

To conclude, our robot can learn to detect doorbell buttons and identify the broken buttons on a portable device in an unsupervised way by haptic exploration. This study focuses on the portable device, which is very common in the human's life. The doorbell buttons were put onto the portable device since they are also representative among kinds of buttons. So, this study should contribute in making robot useful in helping human's life. Moreover, the unsupervised learning algorithm, *k*-means, is used to do the learning work and the robot learns to complete the tasks by self-exploration. These two things make the robot can learn to detect the doorbell buttons and identify the broken ones autonomously and in an unsupervised way. C/C++ codes were wrote to make the detection real-time and the learning on-line since the robot can update the experience database and detect the doorbell buttons and broken ones right after the completion of press procedure.

For the future work, more kinds of buttons (or soft surfaces) except doorbell buttons can be tried. Doorbell buttons work very well in this study because they are kind of hard to press and can generate significant travel distance. Future work may have a try on some other buttons say keyboard's keys which need less external force, and some buttons generating less significant travel distance. Some soft surfaces can also be tried say if the robot will be confused by these kinds of non-button surfaces, which can also generate travel distance. Buttons may not be restricted onto the portable device. They can be on a big board just like the doorbell buttons are on the door. Then in this case, it needs to generate other kinds of press exploratory procedure except the way of only closing/opening one finger. The experiment setup in the previous study then can be used. The robot will go to explore the big board perpendicularly standing in front of the robot to detect the buttons. The press exploratory procedure will be from the movement of the whole arm and should be designed elaborately to be similar to human's press when human press a doorbell buttons.

9. References

J. Miura, K. Iwase, and Y. Shirai 2005. Interactive teaching of a mobile robot. In IEEE International Conference on Robotics and Automation, volume 3, page 3378.

P. Hauf and G. Aschersleben 2008. "Action-effect anticipation in infant action control," Psych. Research, vol. 72, no. 2, pp. 203–210.

Thomaz, A. 2006. *Socially guided machine learning*. Ph.D. Dissertation, Massachusetts Institute of Technology.

Klingbeil, E. and Saxena, A. and Ng, A.Y. 2008. "Learning to Open New doors", Stanford University.

Allen, P., and Roberts, K. 1989. Haptic object recognition using a multi-fingered dextrous hand. In *IEEE International Conference on Robotics and Automation*, 342–347.

Caselli, S.; Magnanini, C.; Zanichelli, F.; and Caraffi, E. 1996. Efficient exploration and recognition of convex objects based on haptic perception. In *IEEE International Conference on Robotics and Automation*, 3508–3513.

Klatzky, R. L.; Lederman, S. J.; and Matula, D. 1991. Imagined haptic exploration in judgments of object properties. *Journal of Experimental Psychology: Learning, Memory, and Cognition* 17:314–322.

Lederman, S. J., and Klatzky, R. L. 1987. Hand movements: A window into haptic object recognition. *Cognitive psychology* 19(1):342–368.

Loomis, J., and Lederman, S. 1986. Tactual perception. In Boff, K.; Kaufman, L.; and (Eds.), J. T., eds., *Handbook of human perveption and performance*, 1–41. New Youk: Wiley.

Okamura, A., and Cutkosky, M. 1999a. Haptic exploration of fine surface features. In *IEEE International Conference on Robotics and Automation*, 2930–2936.

Okamura, A., and Cutkosky, M. 1999b. Haptic exploration of fine surface features. In *IEEE International Conference on Robotics and Automation*, volume 4, 2930–2936.

Okamura, A., and Cutkosky, M. 2001. Feature detection for haptic exploration with robotic fingers. *The International Journal of Robotics Research* 20(12):925.

Okamura, A.; Costa, M.; Turner, M.; Richard, C.; and Cutkosky, M. 2000. Haptic surface exploration. *Experimental Robotics VI* 423–432.

Roberts, K. 1990. Robot active touch exploration: Constraints and strategies. In *IEEE International Conference on Robotics and Automation*, 980–985.

Stansfield, S. 1992. Haptic perception with an articulated, sensate robot hand. *Robotica* 10(06):497–508.

Sukhoy, V., Sinapov, J., Wu, L., and Stoytchev, A. 2010a. "Learning to Press Doorbell Buttons," In Proceedings of the 9th IEEE International Conference on Development and Learning (ICDL), Ann Arbor, Michigan, August 18-21, pp. 132-139.

Sukhoy, V., and Stoytchev, A. 2010b. Learning to detect the functional components of doorbell buttons using active exploration and multimodal correlation. In *IEEE International Conference on Humanoid Robots (Humanoids)*,
572–579.

Sukhoy, V.; Sahai, R.; Sinapov, J.; and Stoytchev, A. 2009. Vibrotactile recognition of surface textures by a humanoid robot. In *The 9-th IEEE-RAS International Conference on Humanoid Robots*, 57–60.

MacQueen, J. B. (1967). "Some Methods for classification and Analysis of Multivariate Observations". Proceedings of 5th Berkeley Symposium on Mathematical Statistics and Probability. University of California Press. pp. 281–297.