Learning to Detect the Functional Components of Doorbell Buttons Using Active Exploration and Multimodal Correlation

Vladimir Sukhoy and Alexander Stoytchev

***Abstract*— This paper describes a large-scale experimental study, in which a humanoid robot learned to press and detect doorbell buttons autonomously. The models for action selection and visual detection were grounded in the robot's sensorimotor experience and learned without human intervention. Experiments were performed with seven doorbell buttons, which provided auditory feedback when pressed. The robot learned to predict the locations of the functional components of each button accurately. The trained visual model was also able to detect the functional components of novel buttons.**

I. INTRODUCTION

Buttons are everywhere around us. There are buttons in every room, in every car, and in every elevator. These simple 1-dof devices are the method of choice for controlling the state of many mechanisms and systems. The average human interacts with dozens of buttons every day without even noticing. By necessity, robots that operate in human-inhabited environments will have to learn how to use buttons. Otherwise, they will not be very useful.

Pressing buttons is still challenging for robots for two reasons. First, buttons are designed for humans, not for robots. Their shapes and sizes are optimized for human fingers. Buttons that are pressable by humans may not be pressable by robots (e.g., because they are too small for robotic fingers). Furthermore, seemingly small differences in finger morphology can lead to significantly different use patterns. For example, some plastic buttons may be too slippery for robotic fingers made of brushed aluminum, forcing the robot to use approach trajectories that rely on the external button housing in order to perform the push. Second, different buttons produce different feedback when their state changes. Some buttons click, others light up, still others provide auditory feedback when pressed. This large variety of buttons makes it impossible to write one program that all robots can use in order to press all possible buttons.

The goal of this work is to develop a framework and a visual model for detecting the functional components of buttons that can be trained from experience. The experiments were performed with seven doorbell buttons that provide auditory feedback when pressed, but the framework can be applied to other buttons as well. For each button the robot performed a large number of oscillatory pushing movements in the vicinity of the button and recorded the auditory, proprioceptive, and visual events that occurred. A visual model was learned, which from the resulting dataset predicted the locations of the functional parts of buttons, i.e., the parts that actually produced sound when pressed. The model was able to generalize to novel buttons.

Developmental Robotics Laboratory, Iowa State University, 3128 Coover Hall, Ames, IA 50011-3060, USA. {sukhoy, alexs}@iastate.edu

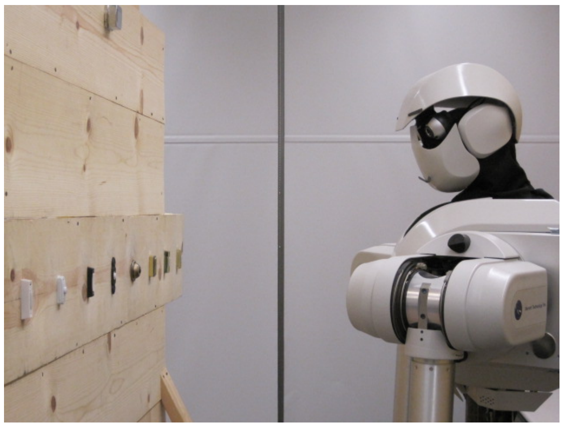


Fig. 1. All experiments were performed with the upper-torso humanoid robot, shown here in front of the buttons.

As the robot explores buttons it keeps track of the location of its fingertip when the buzzer goes off. The spatial density of the auditory events in the robot's visual space is used to train a classifier that maps image patches from visual space to their likelihood of triggering the buzzer. This classifier serves as the robot's model of what the functional component of a button looks like. The model was able to detect the functional components of both familiar and novel buttons.

Finally, the visual model is capable of monitoring its own learning progress in order to estimate the amount of training experience that is required. Specifically, the robot is able to estimate when the predictions of its visual model stop changing. For most buttons this condition is reached after 50-100 trials, which allows our method to learn in real time.

II. RELATED WORK

A. Robotics

The related work on button pushing in robotics can be divided into three main categories as described below.

1) *Pressing buttons is easy, but detecting them is hard.* This category assumes that the physical act of pressing a button is trivial and focuses on the problem of visually detecting the button, which is presumed to be more difficult. The feedback from the button (e.g., click, light, sound) is not commonly used to detect it. In fact, this feedback is typically ignored. Work in this category has focused on detecting elevator buttons [1] [2] [3]. In some of these studies [2] [3], the detection of elevator buttons was improved using the fact that they are often arranged in a grid.

2) *Both pressing and detecting buttons is hard.* The second category assumes that both detecting and pressing buttons, switches, and other small widgets designed for humans is still too difficult for robots. Therefore, they choose to make these widgets more visible and manipulable by attaching tags to them (e.g., reflective markers or RFID tags) [4]. For a button, this tag carries information about how and where a robot can press it and what would be the result of this action. The main focus of this research is on robotic applications in the home that are enabled by these augmentations [5].

3) *Button pressing as a social learning task.* This category has focused on social learning strategies for teaching a robot to press buttons. It was demonstrated that a robot can learn to press a button using human social cues as feedback [6]. The main focus, however, was on interpreting and learning from human social cues and not on the physical task of pressing a button.

Our work differs from previous approaches in the following ways: 1) We treat both the detection task and the pressing task as challenging; 2) The visual model for detecting a button is constructed incrementally from the multimodal events produced by the button (i.e., we don't ignore the feedback that the button generates when it is pressed); 3) Our approach does not rely on a human to tell the robot how and where to press the button.

Because of the different embodiments of the human and the robot, it is not always possible to tell a robot how to press a button. Therefore, the robot learns its own representation by actively exploring the button and detecting the multimodal events resulting from this interaction. As the robot gathers training data it learns a visual model to detect the button. The model's generalization properties improve with additional experience and it can be used to detect novel buttons.

The strategies that the robot used to explore the buttons were formulated in the information-theoretic paradigm of active learning [7] [8] [9]. Active learning has been used successfully for grasping tasks [10] [11] [12] [13]. The exploration strategies are analyzed in more detail in [14], which is based on the same dataset. The focus of this paper is on combining active learning with an ability to generalize the experience obtained from it. The experience is generalized across different modalities: it is obtained primarily using proprioception and audio, but the generalization occurs in the visual domain.

B. Developmental Psychology

In psychology, E.J. Gibson [15] analyzed a large body of research on exploration and knowledge acquisition and concluded that, for a human infant, "observations made possible via both exploratory and performatory actions provide the material for his knowledge of the world – a knowledge that does not cease expanding, whose end (if there is an end) is understanding."

Others have also noted that the experience that infants obtain while exploring objects stimulates their further interest in these objects [16]. 7-11 months old infants are not interested in seeing objects, or people who manipulate objects, until the infants have had the chance to play with these objects [16]. Only after playing with the objects the infants became interested in observing how people manipulate these objects [16]. There is also evidence that human infants can anticipate events that occur during manipulation. In particular, infants as young as 9 months old can predict that an interesting sound will be heard or a bright light will be seen when an experimenter presses a colored button [17].

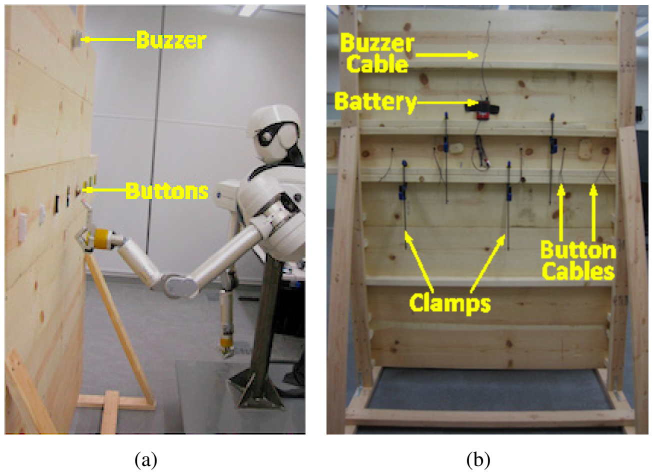


Fig. 2. The experimental setup: (a) the robot exploring button number 4; (b) The back side of the experimental fixture.

III. EXPERIMENTAL SETUP

A. Robot

The button pushing experiments were performed with the upper-torso humanoid robot shown in Fig. 1. The robot used in the experiments had two Barrett Whole Arm Manipulators (WAMs) as arms with two Barrett BH8-262 Hands as end effectors. A color marker was attached to the fingertip of finger F3 on the left hand – the hand used to press the buttons – in order to simplify its visual tracking.

B. Experimental Fixture with Doorbell Buttons

Seven different doorbell buttons were mounted on a wooden fixture as shown in Fig. 2. The middle segment of the fixture could slide horizontally, which allowed different buttons to be placed in front of the robot without moving the fixture or the robot. This was necessary for the proper collection and analysis of such a large-scale dataset. The button currently explored by the robot was connected to a buzzer, which produced loud auditory feedback.

C. Experimental Trials

The robot explored the buttons with 3 different exploration strategies: random, stimulus-driven, and uncertainty-driven (see Section IV-A). For each strategy, the robot performed 200 trials. In addition, an evaluation set of 400 trials was collected for each button, which was used to evaluate the performance of the three strategies. The evaluation set was collected using the random strategy, but the data was used only for testing and not for training. To summarize, for each of the 7 buttons the robot performed $3 \times 200 + 400 = 1000$ trials, for a total of $7 \times 1000 = 7000$ trials. Each trial consisted of 5 pushing behaviors (see Fig. 3), which resulted in a dataset with $35,000$ pushing behaviors.

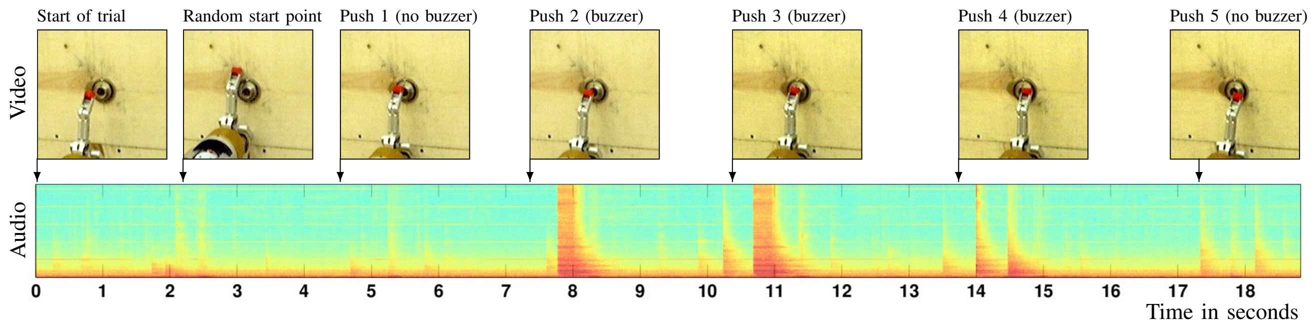


Fig. 3. A sample trial performed by the robot. Video frames for each of the five pushing behaviors are matched to the corresponding sound spectrogram. The robot's field of view is larger than the images shown here, which were cropped to show only the area around the button.

The starting position for each push was the end position of the previous push. The end point of the push was selected based on the learning strategy that was used. To randomize the starting position of each trial, the robot started with a random push that was not counted toward the 5 pushes in the trial. Each trial lasted for approximately 18-20 seconds (the setup time was 3–5 seconds and each of the 5 pushing behaviors took $\sim$3 seconds).

All 1000 trials with each button were performed sequentially without interruptions. It took about 6 hours of exploration to collect all the data for one button (3 exploration strategies plus evaluation set). The total time for collecting the dataset with 7 buttons was about $7 \times 6 = 42$ hours.

This large dataset was necessary to properly analyze the effects of more experience on the generalization abilities of the robot under the three exploration strategies. As described below, the robot learned to press and detect even the most challenging buttons in far fewer trials.

D. Behavioral Parametrization

Each trial was started from a fixed reference arm position, for which the robot's finger was above a button. At the start of each trial, the robot pushed the area around the button at random. This push randomized the start position of the exploration and did not count towards it, even if it happened to trigger the buzzer. Then the robot performed 5 exploratory pushing behaviors in the area around the button. The start position for each of these behaviors was the end position of the previous push. The end position was determined using the current exploration strategy. From the start position, the robot backed up its arm halfway towards the fixed reference position without reaching it and then pushed towards the end position. At the end of the trial the robot returned its arm completely to the reference position and started the next trial.

A vector $x \in \mathbb{R}^6$, which was a concatenation of two unit vectors $x^{(b)}, x^{(p)} \in \mathbb{R}^3$ in the robot's Cartesian space, parameterized each pushing behavior. $x^{(b)}$ was the direction of the backup movement towards the fixed arm reference position. $x^{(p)}$ was the direction of the following push. To provide equal weights for both backward and forward movements, which constitute a pushing behavior, both $x^{(b)}$ and $x^{(p)}$ are movement directions rather than exact positions. The parameters for the pushing behaviors were sampled uniformly in joint space, which resulted in non-uniform sampling in Cartesian and visual space. Even though the reference arm position was the same for all behaviors, the parametrization, and the resulting movement, was different for behaviors with different start and stop positions on the surface of the experimental fixture.

To summarize, the behaviors in each trial resulted in oscillatory finger movements around the button. Infants perform repetitive movements when they learn similar tasks [18].

E. Sensory Data

The robot recorded sensory data using 3 modalities: video, audio, and proprioception. Video was recorded at 10 fps and 640×480 pixels per frame from the Logitech QuickCam Pro 4000 webcam in the robot's left eye. Audio was recorded at 44.1 KHz (16-bit stereo) from the Audio-Technica U853AW Hanging Microphone mounted in the robot's head. Joint angles and joint torques for each of the 7 joints of the left WAM arm were recorded at 500 Hz. Audio and proprioception were processed in real time to detect events and to train the predictive models used by the 3 exploration strategies (see Section IV-A). The data from all modalities was timestamped and stored to disk for additional analysis.

Proprioceptive events were detected when the magnitude of the torque for any of the 7 joints exceeded a predefined threshold. These events were recorded when the robot's finger was "stuck" and could not move any further (e.g., because it was pressing hard against the wall or the button). Whenever a proprioceptive event was detected, the robot interrupted the current movement and started the next one.

Auditory events were recorded when the robot heard the buzzer. The detection was achieved in two steps. First, candidate regions were extracted from the audio stream when the volume exceeded a predefined threshold. Regions shorter than 0.03 seconds were not analyzed. Longer candidate regions were truncated after 0.1 seconds. If the buzzer was heard more than once during a pushing behavior then only the first instance was recorded. Second, the Discrete Fourier Transform was applied to each candidate region to get its spectrogram (with 257 frequency bins and a Hamming window of 512 samples overlapped at 50%). A 20-bin frequency component histogram was calculated from the spectrogram and its bin values were used as input features for a cascade of two classifiers (Naive Bayes and K^* [19]), which made the final decision. The cascade structure was determined from a similar dataset [20], in which candidate regions with and without the buzzer sound were labeled manually.

IV. METHODOLOGY

A. Exploration Strategies

Three different exploration strategies were evaluated: random, stimulus-driven, and uncertainty-driven. Each strategy used a different predictive model M to choose the next pushing behavior from a set of $N = 25$ random candidates, generated from scratch every time. From these random candidates, the current strategy selected one, which was then performed by the robot. The strategies differed in the specific rule that they used to select the winning behavior from the set of random candidates, given the current state of the model.

More formally, as the robot explored a button, it recorded tuples (b_i, o_i) where $b_i \in \mathbb{R}^6$ were the behavioral parameters and $o_i \in \{buzzer, no\ buzzer\}$ was the observed outcome. These tuples constituted experience that the robot obtained from exploration. The predictive model M was trained using this experience to estimate the conditional probability $\Pr(o|b)$ of an outcome o given the candidate behavior b. The three strategies were formulated as follows.

1) *Random Strategy:* always selected the first behavior from the 25 random candidates, i.e., $j = 1$. This strategy did not use a predictive model and the resulting exploration was random.

2) *Stimulus-Driven Strategy:* selected the behavior that was most likely to produce the *buzzer* outcome among the N candidates according to the predictive model M. More formally, the behavior j parametrized by $x_j \in \{x_1, \ldots, x_N\}$ was chosen such that:

$$j = \underset{i=1,\ldots,N}{\operatorname{argmax}} \Pr(buzzer|x_i).$$

3) *Uncertainty-Driven Strategy:* selected the behavior for which the predictive model M was most uncertain about its outcome. For each of the $N = 25$ candidate behaviors, the entropy of the conditional distribution of its outcomes quantified the uncertainty. More formally, this strategy selected behavior j parametrized as $x_j \in \{x_1, \ldots x_N\}$ such that

$$j = \underset{i=1,\ldots,N}{\operatorname{argmax}} \sum_{o \in O} -\Pr(o|x_i)\log(\Pr(o|x_i)).$$

The choice of a mechanism for a predictive model M was motivated by the need to update it incrementally as the robot collected more experience. The results of a pilot study [20] showed that the k-NN classifier performed well for a similar task. The k-NN classifier is data-driven and the computational cost of updates as more training instances become available is low. In this study we set $k = 5$ (an odd k is useful to avoid tied votes), i.e., the classifier estimated $\Pr(o|x)$ from the outcome distribution of the 5 nearest neighbors of x in the robot's experience. For example, if 4 out of 5 of these neighbors had *buzzer* outcomes, the estimate for $\Pr(buzzer|x)$ was 4/5.

B. Spatial Distributions of Auditory Events

As the robot explored a button, it recorded the visual location of its fingertip every time it heard the buzzer. Fig. 6(e) shows the resulting 2D point clouds for the uncertainty-driven strategy. The point clouds for the test set are shown in Fig. 6(b). The apparent scattering of the points is due to the fact that the robot can press a button with parts of its finger other than its fingertip, which was the only thing that was tracked. Fig. 4 shows several examples of the robot successfully pressing a button even though the visual position of its fingertip does not coincide with the functional component of the button.

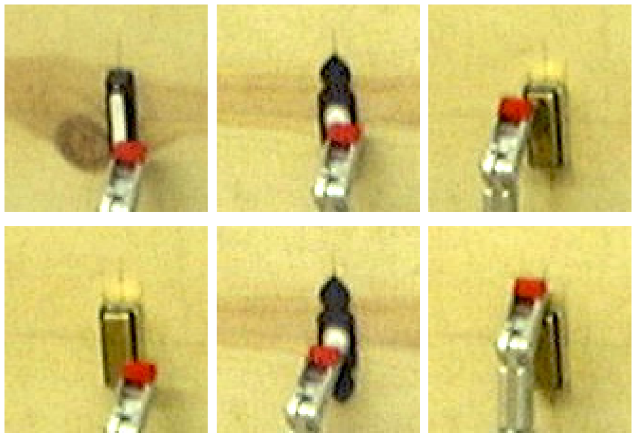


Fig. 4. Snapshots from the dataset showing different ways in which the robot was able to ring the doorbell buttons. Note that the red marker on the robot's fingertip does not necessarily have to overlap with the functional component of the button.

The point clouds offer a good way to visualize the positions of the auditory events in camera coordinates. A better way to visualize this data, however, is to compute the density of the auditory events, shown in Fig. 6(c) and 6(f). These densities were estimated using k-NN with $k = 5$, which is a standard technique [21]. The density of the auditory events contains information that can be used by the robot to decide where a button should be pressed. Furthermore, as described below, the density can be used to learn a visual model of what a button looks like.

C. Visual Features

As the robot explored a button, it attempted to simultaneously build a visual model that could detect the functionally meaningful part of the button (i.e., the part that produces auditory feedback when pressed). The robot extracted features from the color image using the feature detector described in [22]. These features have been used successfully by a robot to detect grasp points in novel objects, which was one reason why they were selected for this task.

The features aim to extract texture, edge, and low-frequency color information from the original RGB image, which is converted to YC_bC_r before further processing. Texture is extracted from the Y image, which is convolved with nine 3×3 Laws' masks. Edge information is extracted also from the Y image by convolving it with six 5×5 edge filters. Finally, low-frequency color information is obtained from the C_b and C_r images, which are convolved with the first Laws' mask. Fig. 5 shows this procedure, which results in 17 images. These images are squared before the next operations are performed on them to extract the features.

The original image is then partitioned into 10×10 pixel patches and a 459-dimensional feature vector is computed for

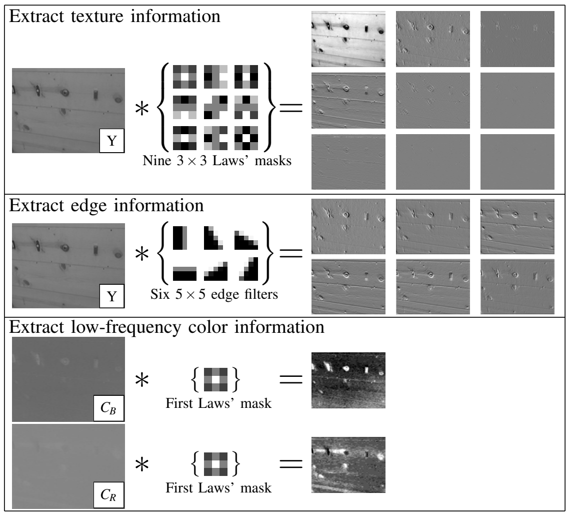


Fig. 5. Illustration of the convolution step that is used by the feature detector. The RGB image is converted to YC_bC_r and the three channels are convolved with different convolution masks. The resulting 17 images are used to compute the features. For the purposes of visualization, the convolution results were clamped at the 1st and the 99th percentile.

each patch as follows. For a patch p, the first 17 components of the feature vector are calculated by summing up the 100 pixel values of the corresponding 10×10 pixel patches in the 17 convolved images. This procedure is repeated for the 24 neighbors of p in a 5×5 patch window centered at p. This results in another 24×17 features (computed from a 50×50 pixel window in the original image).

Two extra spatial scales that capture more global properties of the image are included in the feature vector as well. For the first scale, the original 640×480 RGB image is zoomed out by 3x. This results in a 214×160 pixel image, which is convolved with the 17 convolution masks shown in Fig. 5. For the patch of interest, p, in the original image, corresponding patches of size roughly equal to 10×10 pixels are identified in the 17 convolved images for the zoomed out image. Their pixel values are summed up to produce 17 features. The same procedure is repeated for the second spatial scale, which uses a zoom out factor of 9x. For more details, see [22] and the corresponding source code.

To summarize, the number of features computed for every 10×10 pixel patch in the original image is:

$$\left(\underset{\text{patch}}{1} + \underset{\text{neighbors}}{24} + \underset{\text{scales}}{2}\right) \times \underset{\text{convolutions}}{17} = \underset{\text{features}}{459}.$$

D. Learning a Visual Model for a Button

The visual model was implemented as a Logistic Regression-based classifier [23] that mapped visual features to a binary class variable [22]. In other words, given a 10×10 pixel patch and its corresponding 459-dimensional feature vector, the classifier predicted whether the patch belonged to a functional component or not. The classifier model was learned only from training instances grounded in the robot's own experience with the buttons. Feature vectors for training the model were computed only for the image patches explored by the robot. For the purposes of training, a patch was marked as belonging to a functional component if the average spatial density of auditory events for this patch was greater than $\mu + 3\sigma$ (where the mean, μ, and the standard deviation, σ, were computed using all explored patches for a given button). Otherwise, the patch was marked as not belonging to a functional component.

While it is easy to specify what is a button, it is much harder to specify what is not a button. Because the robot recorded only images of buttons during the exploration process, training the classifier required negative examples in order to improve its generalization performance. These additional negative examples were provided from synthetic images, in which the pixels were filled with uniform random noise. This is based on the assumption that pixels filled with uniform random noise cannot represent anything meaningful in the robot's environment.

V. RESULTS

A. Performance Measures

The performance was measured on the task of predicting whether a given 10×10 pixel patch in the image of the button belongs to a functional component for this button or not. The visual model described in Section IV-D was used to generate these predictions given the visual features calculated for this patch. The predictions were compared to the ground truth for these patches, which was obtained using the data from the evaluation set.

The typical size of the exploration region for a single button in visual space was 110×120 pixels, or 11×12 patches. In contrast, the functional component of a typical button occupies less than 10 patches. A naive strategy can mark all 132 patches as containing no functional components, achieving accuracy of about 92.4%. Even though the accuracy is high, the predictions are useless. Thus, raw accuracy is an inadequate performance measure in this case. A better measure is the Cohen's kappa statistic [24], which takes the probability of chance agreement into account:

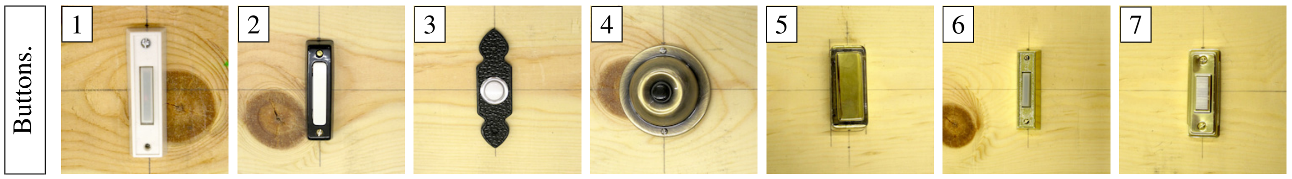
$$\kappa = \frac{\Pr(a) - \Pr(e)}{1 - \Pr(e)},$$

where $\Pr(a)$ is the probability of a correct prediction by the model and $\Pr(e)$ is the probability of a correct prediction by chance. In our example, $\Pr(a) = \frac{122}{132}$ and $\Pr(e) = \Pr(a)$, so $\kappa = 0$. Now, suppose that there were 6 true positives, 1 false positive, 4 false negatives, and 121 true negatives. Then $\Pr(a) = \frac{127}{132}$, $\Pr(e) = \frac{10}{132} \cdot \frac{7}{132} + \frac{122}{132} \cdot \frac{125}{132} \approx 0.879$, and $\kappa \approx 0.686$, which is a more useful performance measure.

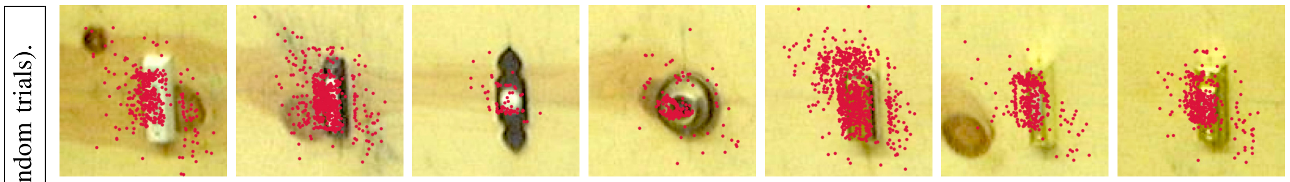
Fig. 7(a) shows the raw accuracy for all three exploration strategies (averaged over all 7 buttons). For reasons described above, the accuracy is very high even with little training. The corresponding values for the kappa statistic are shown in Fig. 7(b). Fig. 6(h) shows the kappa values for individual buttons. The robot's learning performance is very good (results with $\kappa \geq 0.8$ are considered very strong). For further discussion on different kappa-like measures and their characteristics the reader is referred to [25].

B. Evaluating the Visual Model

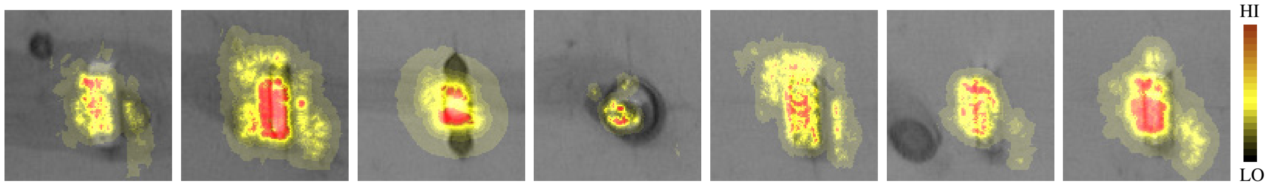
To evaluate the visual model we obtained "ground truth" measurements for all buttons. This was done by collecting a



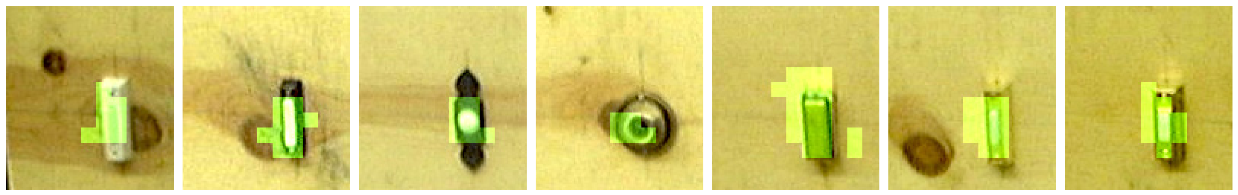
(a) The seven doorbell buttons explored by the robot.



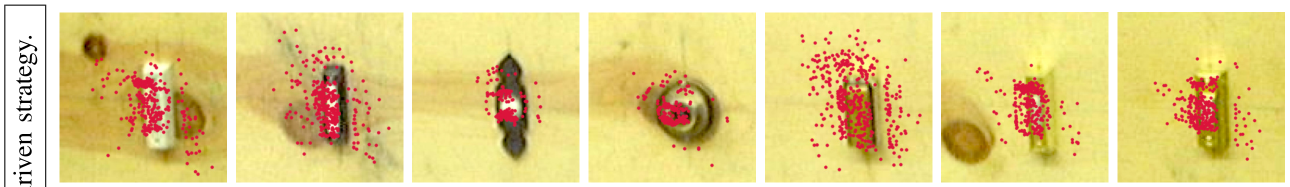
(b) Auditory events localized in visual space. Each point corresponds to the location of the robot's finger when the buzzer went off.



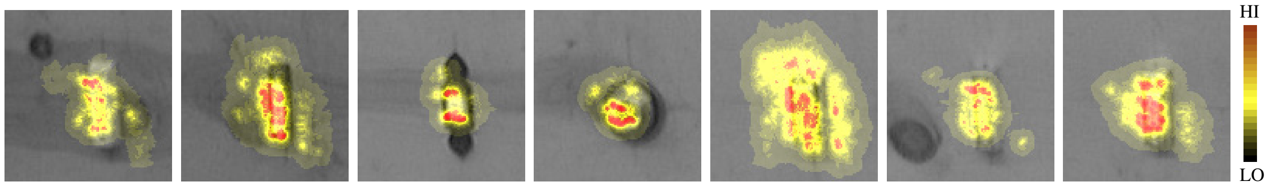
(c) Densities for the auditory events in visual space shown in (b), estimated using k-NN with $k = 5$.



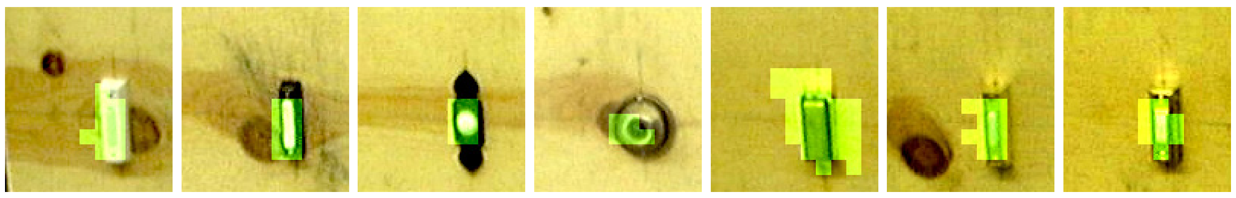
(d) "Ground truth" about the visual positions of the functional components extracted by thresholding the densities shown in (c).



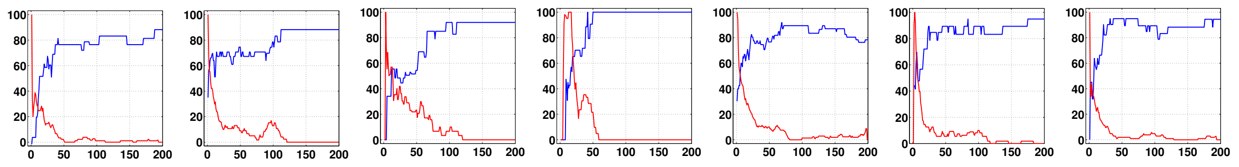
(e) Auditory events localized in visual space. Each point corresponds to the location of the robot's finger when the buzzer went off.



(f) Densities for the auditory events in visual space shown in (e), estimated using k-NN with $k = 5$.

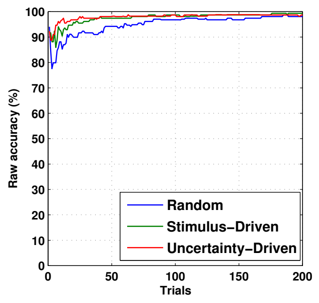


(g) Functional components for each button learned after 200 trials performed with the uncertainty-driven strategy.

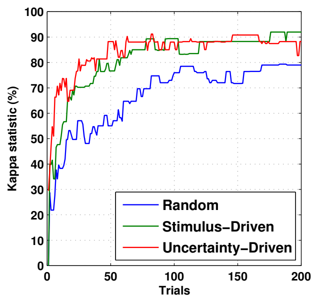


(h) Two measures of learning progress. The predictions after 200 trials are shown in (g). The "ground truth" is shown in (d).
The kappa statistic(blue line, %) and the normalized smoothed rate of change(red line, %) are shown as functions of the number of trials.

Fig. 6. Summary of the experimental results for a familiar button. See text for more details.



(a) Raw accuracy



(b) Kappa statistic

Fig. 7. Results for predicting whether a 10×10 pixel patch in an image belongs to the functional component of a familiar button as a function of the number of trials (average over all 7 buttons).

test set of 400 trials, each with 5 random pushing behaviors. The locations of the auditory events in visual space for the test set are shown in Fig. 6(b). The density of these events in visual space is shown in Fig. 6(c). The densities were estimated from the points shown in Fig. 6(b) using k-NN with $k = 5$. The densities were then thresholded as described in Section IV-D and the high-density image patches were labeled as belonging to a functional component. Figure 6(d) shows these 10×10 image patches highlighted in green.

The performance of each exploration strategy was quantified by comparing its classifier model predictions against the "ground truth." Figure 6(g) shows the final predictions after 200 trials with the uncertainty-driven strategy for a familiar button. The auditory events and their densities are shown in Fig. 6(e) and Fig. 6(f). The kappa statistic and the normalized rate of change are shown in Fig. 6(h). For each button, the best performance with this strategy is obtained in 50-100 trials. The final predictions after 200 trials match the ground truth very well ($\kappa \approx 0.8$ or above). This can be confirmed by visually comparing Fig. 6(d) and Fig. 6(g).

Figure 7 shows the median performance of the visual models when tested on familiar buttons as a function of the number of trials (the plots are averaged over all seven buttons). Both active learning strategies outperform the random strategy. The uncertainty-driven strategy is the fastest learning strategy. As more learning trials are performed, the performance of all strategies improves while the distinctions between them become less pronounced. This is to be expected, because when more and more data becomes available their visual models become very similar (they are based on the same logistic regression classifier).

The visual models were also tested on images of novel buttons. To do this the models were trained on 6 buttons and tested on the remaining one. Figure 8 shows the prediction results for button number 7 after 5, 20 and 50 training trials with the uncertainty-driven strategy. For comparison, the predictions in the familiar condition are shown as well. As the robot gains more experience its visual model zooms in on the functionally meaningful part of the button. The results for the other two exploration strategies were similar.

Finally, we tested the visual model trained with the data for all 7 explored buttons on completely novel pictures of the buttons. Fig. 9 shows that the robot was able to identify

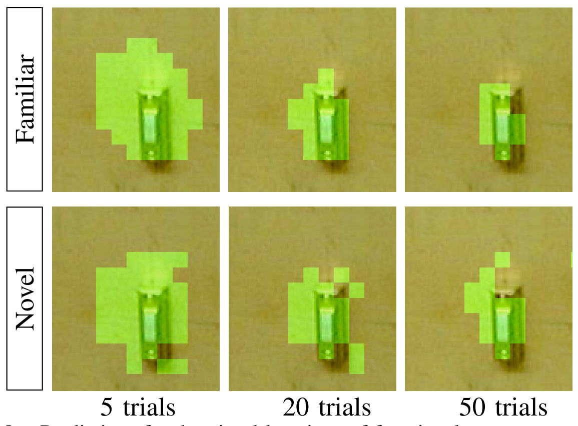


Fig. 8. Predictions for the visual locations of functional components after different amounts of training using the uncertainty-driven strategy. For the novel button, predictions are made using the data collected with each of the remaining 6 buttons over 5, 20, and 50 trials.

the buttons in these pictures. There are a few false positives, which can be attributed to the limited amount of experience. In particular, the negative examples used to train the model were not sufficient to eliminate all false positives.

C. When to Stop Exploring?

This section describes how the robot can estimate its own learning progress and decide when to stop exploring a given button. One possible solution is to stop exploring when the predictions of the robot's visual model stop changing. To quantify this approach, the rate of change of the predictions was computed. More formally, the rate of change is the square root of the number of 10×10 pixel patches for which the class labels changed from one trial to the next. The class labels were computed by thresholding the density of auditory events in visual space (see Section IV-D).

The results for the seven explored buttons are shown in Fig. 6(h). The rate of change curves were smoothed using a running average filter over a window of 20 trials. As the learning performance measured by the kappa statistic increases the rate of change decreases. Unlike the kappa statistic, however, the rate of change can be computed by the robot without the "ground truth" (see Fig. 6(d)). Thus, it can be used as a criterion for deciding when to stop the exploration process. The robot can stop exploring a button when the rate of change is zero or close to zero. For most buttons this condition is true after 100 trials (see Fig. 6(h)).

VI. CONCLUSIONS AND FUTURE WORK

This paper described a framework that a robot can use to learn both how and where to press buttons. The framework was tested on doorbell buttons, which provide auditory feedback, but it can easily be applied to other types of buttons. The representations learned by the robot were grounded in its own experience with the buttons. Unlike previous work, our approach did not require humans to label buttons in the robot's perceptual world in order for it to solve this task.

To detect the functional components, the robot learned a visual model from the spatial and temporal correlations between auditory events and visual percepts. The model

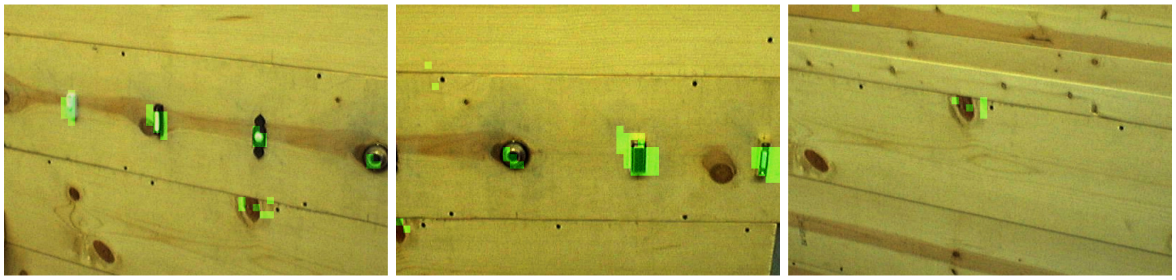


Fig. 9. After the visual model was trained with all 7 buttons, the robot was tested with images of the experimental fixture that it had never seen before.

was tested on the tasks of predicting the functional components of: 1) familiar buttons, and 2) novel buttons. For familiar buttons, the performance was close to perfect. For novel buttons, the model was still able to find a decent approximation based on the experience obtained with other buttons. For both novel and familiar buttons, the performance improved as the robot obtained more experience. In a way, the trained visual model acted like an affordance detector as it was able to label patches in as "pushable there".

The visual model can also be used by the robot to decide when to stop exploring a button. The rate of change in the predictions of the model can serve as a measure of learning progress. The robot can stop the exploration when the predictions of the model stop changing. In other words, the can could stop when pressing the button any further would not help improve the visual model for detecting it.

In general, we found that the button pressing task is neither too easy nor too difficult. Our robot learned to press buttons in 50-100 trials. The visual model for what a button looks like was also learned (and converged) in 50-100 trials. It performed well for both familiar and novel buttons.

Future work can add feedback from touch sensors to the framework, which already supports audio and video. The next generation BarrettHand (BH8-280), which has 96 tactile sensors, should make this a feasible option for our robot. Another possibility is to include the physical resistance of a button as one of its properties. Currently we are exploring whether the resistance of a button during its small travel distance (1- 2mm) can be detected reliably by the WAM.

At present, the exploration strategies do not take into account the predictions of the visual model. The framework can be extended to incorporate the patch-based representation into the selection of the next pushing behavior.

The mechanism for detecting events in the current framework is engineered for very specific types of auditory and proprioceptive events: the buzzer sound and high joint torques. The events are mapped to vision using the color tracking of a marker on the robotic fingertip. Future work can replace these mechanisms with more general models for detecting, categorizing, and matching events in different modalities with the robot's actions.

Finally, the framework can be extended to handle other small widgets, including: light switches, elevator buttons, sliders, and levers. One possible challenge task for a robot would be to learn how to manipulate the widgets on a novel instrument panel.

References

[1] J. Miura, K. Iwase, and Y. Shirai, "Interactive teaching of a mobile robot," in *Proc. of ICRA*, 2005, pp. 3378–3383.

[2] K.-T. Song and T.-Z. Wu, "Visual servo control of a mobile manipulator using one-dimensional windows," in *Proc. of Industrial Electronics Society*, vol. 2, 1999, pp. 686–691.

[3] E. Klingbeil, B. Carpenter, O. Russakovsky, and A. Ng, "Autonomous operation of novel elevators for robot navigation," in *Proc. of ICRA*, 2010, pp. 751–758.

[4] R. Katsuki *et al.*, "Handling of objects with marks by a robot," in *Proc. of IROS*, vol. 1, 27-31 2003, pp. 130 – 135.

[5] H. Nguyen, T. Deyle, M. Reynolds, and C. Kemp, "PPS-tags: Physical, Perceptual and Semantic tags for autonomous mobile manipulation," in *Proc. of the IROS Workshop on Semantic Perception for Mobile Manipulation*, 2009.

[6] A. Thomaz, "Socially guided machine learning," Ph.D. dissertation, Massachusetts Institute of Technology, 2006.

[7] A. Baranes and P.-Y. Oudeyer, "R-IAC: Robust intrinsically motivated active learning," in *Proc. of ICDL*, 2009.

[8] A. Barto, "Intrinsically motivated learning of hierarchical collections of skills," in *Proc. of ICDL*, 2004, pp. 112–119.

[9] F. Kaplan and P.-Y. Oudeyer, "Motivational principles for visual know-how development," in *Proc. of the 3rd Intl. Workshop on Epigenetic Robotics*, no. 101. Lund Univ. Cognitive Studies, 2003, pp. 73–80.

[10] M. Salganicoff, L. Ungar, and R. Bajcsy, "Active learning for vision-based robot grasping," *Machine Learning*, vol. 23, no. 2-3, pp. 251–278, 1996.

[11] L. Montesano and M. Lopes, "Learning grasping affordances from local visual descriptors," in *Proc. of ICDL*. Los Alamitos, CA, USA: IEEE Computer Society, 2009, pp. 1–6.

[12] A. Morales and E. Chinellato, "Active learning for robot manipulation," in *Proc. of the European Conference on AI*, 2004, pp. 905–909.

[13] O. Kroemer, R. Detry, J. Piater, and J. Peters, "Active learning using mean shift optimization for robot grasping," in *Proceedings of IROS*, 2009, pp. 2610–2615.

[14] V. Sukhoy, J. Sinapov, L. Wu, and A. Stoytchev, "Learning to press doorbell buttons," in *Proc. of ICDL*, 2010, pp. 132–139.

[15] E. Gibson, "Exploratory behavior in the development of perceiving, acting, and the acquiring of knowledge," *Annual review of psychology*, vol. 39, no. 1, pp. 1–42, 1988.

[16] P. Hauf, G. Aschersleben, and W. Prinz, "Baby do-baby see!: How action production influences action perception in infants," *Cognitive Development*, vol. 22, no. 1, pp. 16 – 32, 2007.

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

[18] J. Piaget, *The origins of intelligence in children*. Intl. Univ. P., 1952.

[19] J. Cleary and L. Trigg, "K*: An instance-based learner using an entropic distance measure," in *Proc. of ICML*, 1995, pp. 108–114.

[20] L. Wu, V. Sukhoy, and A. Stoytchev, "Toward learning to press doorbell buttons," in *Proc. of AAAI*, 2010, pp. 1965–1966.

[21] R. Duda, P. Hart, and D. Stork, *Pattern classification*, 2nd ed. Wiley, 2001.

[22] A. Saxena, J. Driemeyer, and A. Y. Ng, "Robotic grasping of novel objects using vision," *The International Journal of Robotics Research*, vol. 27, no. 2, pp. 157–173, 2008.

[23] S. le Cessie and J. van Houwelingen, "Ridge estimators in logistic regression," *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, vol. 41, no. 1, pp. 191–201, 1992.

[24] J. Cohen, "A coefficient of agreement for nominal scales," *Educational and Psychological Measurement*, vol. 20, no. 1, pp. 37–46, April 1960.

[25] K. Gwet, *Handbook of Inter-Rater Reliability*, 2nd ed. Advanced Analytics, 2010.