Learning from Unscripted Deictic Gesture and Language for Human-Robot Interactions

Cynthia Matuszek, Liefeng Bo, Luke Zettlemoyer, Dieter Fox
cynthia | lfb | lsz | fox@cs.washington.edu
Computer Science & Engineering, University of Washington
Box 352350, Seattle, WA 98195

Abstract

As robots become more ubiquitous, it is increasingly important for untrained users to be able to interact with them intuitively. In this work, we investigate how people refer to objects in the world during relatively unstructured communication with robots. We collect a corpus of deictic interactions from users describing objects, which we use to train language and gesture models that allow our robot to determine what objects are being indicated. We introduce a temporal extension to state-of-the-art hierarchical matching pursuit features to support gesture understanding, and demonstrate that combining multiple communication modalities more effectively capture user intent than relying on a single type of input. Finally, we present initial interactions with a robot that uses the learned models to follow commands.

1 Introduction

As robots move into the world, the importance of enabling untrained users to interact with them in a natural way increases. While it is possible to teach someone how to use a robot for specified tasks (e.g., with a lexicon of predefined gestures), a more intuitive robot would learn how users naturally use speech and gesture. In turn, having access to the physical world allows for learning to understand natural language pertaining to that world, such as descriptions of visual percepts; this is the *grounded language acquisition* problem (Mooney 2008).

This work demonstrates progress towards a robot that can learn to understand unscripted directives from end-users, and perform operations based on that understanding. We introduce a system that learns to interpret user intent in a physical workspace. Particularly, the goal is to learn to understand the subset of deictic interactions called *directing-to*, which are attention-focusing: language and gesture in which the user is drawing attention to specific objects. We have collected an RGB-D corpus of interactions from users identifying objects in a space, which was then used to train models of language, deictic gesture, and visual attributes.

The language and gesture in this corpus are unscripted, i.e., not predefined by us; people did what came naturally during training and in our evaluation. This goal – training a classifier intended to interpret any arbitrary deictic gesture in a tabletop manipulation setting – is qualitatively different from gesture recognition, in which a set of known gestures must be disambiguated. In order to successfully interpret gestures in this (sometimes quite complex) time-series data, we introduce a novel temporal extension to state-of-the-art hierarchical matching pursuit (HMP) features (Bo, Ren, and Fox 2011). Finally, we demonstrate the application of those models to the task of interacting with indicated objects. (Fig. 1).

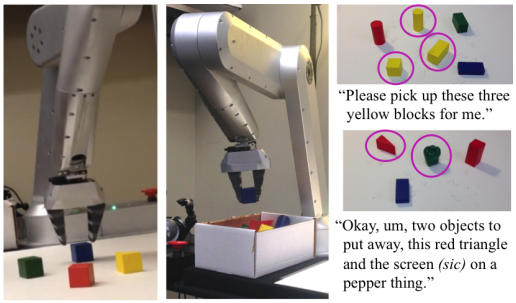

Figure 1: The integrated system. (a) and (b): the Gambit platform picking up a toy and placing it in the bin; (c) examples of the target objects in test scenes, with language used by our test participants (including speech recognition errors).

In the remainder of this paper, we demonstrate that our approach learns from examples provided by untutored users, and that this combination of modalities performs well at capturing users' intent. In order to determine whether the problem area is well-formed, we also evaluate how well humans are able to perform the task, and report on the accuracy of our initial system in a small trial. Our contributions include: A recognition model that combines language, gestures, and visual attributes; a dataset with 364 annotated videos of people describing objects; a prototype implementation that can be instructed using voice and gesture; and a novel technique for unsupervised learning of features rich enough to enable accurate recognition of deictic human gesture.

2 Related Work

Both gesture and language have been studied extensively as possible elements of human-robot interaction (Goodrich and Schultz 2007; Mitra and Acharya 2007). Gesture has been found to be a comfortable, effective modality for teaching robots (Rouanet, Danieau, and Oudeyer 2011), and there is substantial work on gesture recognition in robotics (Wachs et al. 2011), including those using low-cost RGB-D sensors (Ramey, González-Pacheco, and Salichs 2011) and those using temporal or trajectory-based recognition (Nickel and Stiefelhagen 2007; Yang, Park, and Lee 2007). However, rather than performing *recognition* of a gesture, our system is trained to do gesture *understanding* – that is, interpretation of any indicative gesture a user wishes to make.

Teaching robots about objects and attributes of objects in the world is an active target in human-robot interaction (HRI) research. While the specific task we target in this paper uses a limited set of objects and attributes, understanding how people refer to objects in the world is an important element of natural HRI (Sandygulova et al. 2012), and has been studied from the perspective of both human dialogue (Peltason et al. 2012) and vision (Farhadi et al. 2009).

We focus on deictic language and gesture – specifically, those that (Clark 2003) call *directing-to*, which are attention-focusing. There exists work on generating understandable, appropriate gestures, (Liu et al. 2013) integrating very constrained natural language with deictic gesture (Perzanowski et al. 2001), and on how robots may be controlled using gesture (Suay and Chernova 2011). Our work is similar to that of (Obaid et al. 2012), but rather than trying to design a better lexicon of control gestures, our aim is to understand *any* such indicative gesture.

The language learning component of this work fits into the general category of grounded language acquisition (Mooney 2008): The interpretation of human language into semantically informed structures in the context of perception and actuation. This has shown successes in tasks such as robot navigation (Matuszek et al. 2012b) and forklift operation (Tellex et al. 2013). Our learning of attributes is most similar to that of Matuszek and FitzGerald et al. (Matuszek et al. 2012a); however, we collect spoken, rather than typed, language.

Another approach to embodied language learning is to learn situated meanings for symbols from a more emergent, cognitive approach. One way to study embodied acquisition of language is to consider how infants (McGregor et al. 2009; Vollmer et al. 2009) learn symbols and the meanings of gestures; another is to apply the same evolutionary concepts of language to a robotics platform and study the resulting behavior (Steels 2001; Steels and Loetzsch 2012). While these approaches offer the potential of eventually reaching human-level language learning and understanding, to the best of our knowledge, they do not address the question of building a more special-purpose, explicitly robotic learner.

For RGB-D video collection, we use the widely-available Microsoft Kinect sensor. We take advantage of work in the computer vision community on the study of object identification and grounded understanding. In particular, our approach to analyzing gestures is an expansion of Hierarchical Matching Pursuit features, which have shown state of the art performance on static problems such as image classification (Bo, Ren, and Fox 2011) and object identification (Sun, Bo, and Fox 2013; Lai et al. 2013).

3 Problem Statement and Data Collection

The goal of our work is to build a robot that understands the intentions of users without requiring specialized user training. We recorded examples of people describing objects; that data set was used to train models of interactions, which are then used to implement a proof of concept system in which a robot interacts with objects according to user instructions.

3.1 Data Collection and Corpus

Our task is to build a system that understands someone who is indicating objects in a workspace, by verbally describing any combination of object attributes, by gesturing, or both. For each scenario – consisting of RGB-D video and speech – we want to identify positive (indicated) objects in the scene. In order to collect information about how people refer to objects when given few constraints, we used the Kinect sensor (mounted on either a tripod, or the Gambit manipulation platform) to record people describing objects.

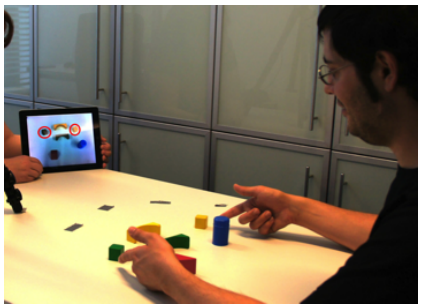

Figure 2: A data collection participant indicating objects. The experimenter shows an iPad with target objects circled in order to avoid linguistic or gestural priming.

Participants were instructed to distinguish objects from a scene, using language and gesture "as if they were describing those objects to a robot", but not given a predefined set of gestures or instructions to use. Experimenters specified what objects to indicate (see Fig. 2) in both training and testing. The data set contained 234 scenes in which two or more objects were to be indicated. Language from the corpus collected was hand-transcribed, although in our robot experiments we use speech recognition. We do not temporally align language with gesture, instead analyzing the entire time-series for deictic motions; this approach presents a noisier learning problem, but is consistent with our data, in which gestures do not always correspond temporally to language.

Scenes were designed to collect data for objects with different spatial relations and combinations of attributes. The length of responses from participants ranged from very brief (around three seconds) to more than a minute. Thirteen participants described each scene. The resulting data set contains examples of language used without gesture, gesture

paired with non-descriptive language (e.g., "These objects"), and gesture and language used together (examples can be seen at: http://tiny.cc/Gambit14).

4 Approach

To accomplish the goal of identifying objects from whatever interactions a user offers, we first train individual components of the system on our training corpus. We assume users might refer to objects by verbally describing any combination of color and shape attributes, by gesturing, or both. Accordingly, for each object, we combine whether it is gestured to and whether it is referred to verbally into a final per-object score, which is used to determine what objects in a scene a person wishes to indicate. The pipeline is as follows:

1. Raw sensor input is processed to identify the workspace, objects, and hands, and language is processed into text.
2. If hands are present, a gesture classifier is used to obtain a probability that each object was gestured to.
3. Vision classifiers return probabilities for the color and shape of each object.
4. Language is combined with the output of the visual attribute classifiers to determine whether each object is referred to in speech.
5. The results of gesture and language analysis are combined into a final *object score* for each object, representing whether the system believes it is a positive object (that is, was somehow indicated by the user).

We describe the individual classifiers used, then discuss how their outputs are synthesized into an evaluation of user intent. Finally, we describe applying the combined system on a robotic manipulation platform.

4.1 Point-Cloud Processing

In this first step, our system extracts the user's hands and the objects from each frame of the RGB-D video. To automatically segment individual objects $o \in O$ from each scene, RANSAC plane fitting is used to remove the table plane, then connected components (segments) of points above that plane are extracted. Remaining points are segmented into clusters (Fig. 5 shows an example of a segmented object).

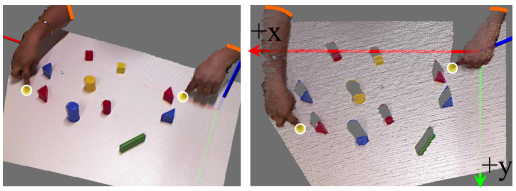

Figure 3: A frame from the data corpus, cropped to the workspace (left), then ***x,y*** axis-aligned to the table (right). Moving clusters of points that extend beyond the workspace are hands (boundaries in orange). The leading-edge point on each hand (yellow) is used to calculate gesture features.

We identify gestures using spatial relationships between the user's hands and the objects over time. For each frame in a scene, we identify the leading edge of a hand (the point on the hand that is furthest from the user's body, from the viewpoint of the interlocutor, see Fig. 3). We then compute the distances between this point and an object. These distances are computed along each of the three coordinate axes and as a single Euclidean distance, resulting in a four-dimensional distance vector at each timestep.

For this work, we use the hierarchical matching pursuit (HMP) approach of (Bo, Ren, and Fox 2011), which uses the matching pursuit encoder to build a feature hierarchy layer by layer, rather than relying on hand-crafted features. HMP has proven to be on par with state of the art for object recognition, scene recognition, and static event recognition. In this work, we explore whether it can be extended to dynamic classification problems.

4.2 Gesture Classification

Once the user's hands and the objects on the table are extracted for the complete video sequence, we try to determine for each object whether the user is gesturing to that object in a way meant to indicate it. This task turns out to be quite complex, due to the substantial variability in how people use deictic gesture (Clark 2005); Fig. 4 shows some examples.

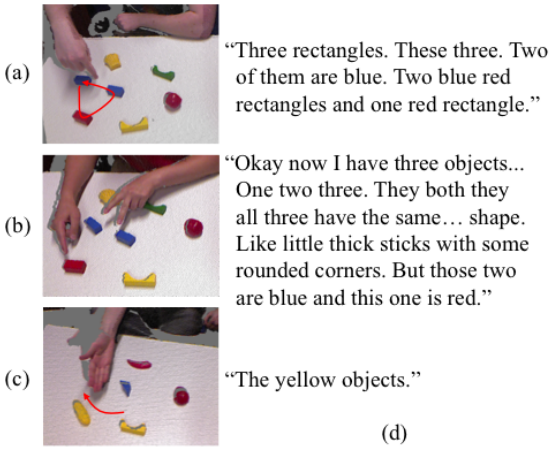

Figure 4: Examples of unscripted gesture and language. (a) A circular pointing motion looping around the objects; (b) pointing with multiple fingers and both hands; (c) an open-handed sweep above objects. (d) gives examples of collected language for the two scenarios shown. The grammatical errors are typical of speech.

For temporal reasoning, we adapt the HMP architecture, which performs pooling over the temporal sequence of the gesture, rather than the 2D grid of an image as in (Bo, Ren, and Fox 2011). The key novelty is in introducing a binary encoding that maintains important scale information. As described above, for each object, we extract a sequence of spatial features from the video: the distance of the hand's leading edge from that object, and the hand's direction relative to the centers of objects in x, y, and z, giving a four dimensional vector at each timestep. Rather than applying a classifier directly to this sequence, we use *sparse coding*, which

has become a popular tool in several fields, including signal processing and object recognition (Yang et al. 2009) – to learn rich features that are more suitable for the classification task.

Sparse coding is an approach to finding rich representations of an input signal Y; from unlabeled input data, functions are discovered that capture higher-level features in the data, making it unnecessary to manually model those features. More specifically, sparse coding models data as sparse linear combinations of codewords selected from a codebook D, which is trained using the efficient dictionary learning algorithm K-SVD. The key idea is to learn the codebook: a set of vectors, or *codewords*. Data can then be represented by a sparse, linear combination of these codebook entries.

Since our gesture recognition task entails time-series data of no fixed length, we operate over the collection of four-dimensional vectors from the entire time-series T, the sequence of frames in which a person was gesturing. Here, we use the efficient dictionary learning algorithm K-SVD and orthogonal matching pursuit to build high-level features.

K-SVD finds the codebook $D = [d_1, \cdots, d_M] \in R^{H \times M}$ and the associated sparse codes $X = [x_1, \cdots, x_N] \in R^{M \times N}$ from a matrix $Y = [y_1, \cdots, y_N] \in R^{H \times N}$ of observed data by minimizing the reconstruction error:

$$\min_{D,X} \|Y - DX\|_F^2 \quad (1)$$
$$s.t.\ \forall m,\ \ \|d_m\|_2 = 1 \ \text{ and } \ \forall n,\ \ \|x_n\|_0 \leq K$$

where H, M, and N are the dimensionality of codewords, the size of codebook, and the number of training samples, respectively. $\|\cdot\|_F$ denotes the Frobenius norm, while the zero-norm $\|\cdot\|_0$ counts non-zero entries in the sparse codes x_n. K is the sparsity level controlling the number of non-zero entries.

Classic K-SVD normalizes the L_2 norm of each codeword to be 1, so learned codebooks don't capture magnitude information of input data. This is a useful property for image recognition, as spatial pooling and contrast normalization over sparse codes can generate features robust to lighting condition changes (Bo, Ren, and Fox 2011). However, magnitude information is critical for gesture recognition. To allow codewords to encode magnitude information, we remove normalization constraints and limit sparse codes to be binary values:

$$\min_{D,X} \|Y - DX\|_F^2 \quad (2)$$
$$s.t.\ \forall n,\ \ x_n \in \{0,1\}^M \ \text{ and } \ \|x_n\|_0 \leq K$$

We design a K-SVD-like algorithm to decompose the above optimization problem into two subproblems, encoding and codebook updating, which are solved in an alternating manner. ENCODING: At each iteration, the current codebook D is used to encode the data input Y by computing the sparse code matrix X, using matching pursuit (Mallat and Zhang 1993). UPDATE: Then, the codewords of the codebook are updated one at a time by gradient descent optimization, giving a new codebook. The new codebook is used in the next iteration to recompute the sparse code matrix followed by another round of codebook updating, repeated to maximum iterations.

With the learned codebook, we are able to generate features representing the whole gesture sequence. Since a gesture that points to a specific object could occur in any timestep, we perform temporal max-pooling by maximizing each component of binary sparse codes of the four dimensional vectors over all timesteps, thereby generating features robust to temporal changes:

$$f_G = \left[\max_{j \in T} |x_{j1}|, \cdots, \max_{j \in T} |x_{jM}|\right] \quad (3)$$

We run logistic regression over these features to train a classifier, which can be run over a new video sequence of gestures and objects. For each object o, this classifier gives h_o, the probability that object o is being gestured to:

$$P(h_o|\Theta^P) = \frac{e^{\Theta_g^P}}{1 + e^{\Theta_g^P}} \quad (4)$$

where Θ_g^P are the parameters in Θ^P for the gesture classifier.

4.3 Color and Shape Classification

Our goal is to determine what attributes each object in a scene has. More formally, for each color and shape present in our corpus, we want to return a probability that each object has that attribute. We let C and S be sets of discrete symbols which denote known colors and shapes, respectively:

$C = \{$*blue,green,red,yellow*$\}$

$S = \{$*arch,ball,cube,cylinder,rectangle,triangle,other*$\}$

Because the focus of this work is on how gesture can be interpreted and used with language, we use a simple set of attributes; however, note that we are using a lower resolution RGB-D camera and trying to learn classifiers that correspond to noisy language (for example, an apple, a potato, and a pepper all referred to as "round"). Any classifier could be used in the same framework, and we intend to incorporate additional classifiers. We also make the assumption that the objects we are working with are separated on a planar surface detectable using RANSAC.

To perform attribute classification, we use a similar method to the work described in classifying gestures, without temporal extensions. At a high level, the goal is to find a set of higher-level features from the raw camera inputs, which give good predictive power for our attributes. This means training a binary classifier for each attribute. On each object, we extract HMP features, and the training inputs Y are tiled sub-squares of the images.

Here, spatial max pooling instead of our temporal max pooling is applied with the learned codebooks to aggregate the sparse codes. These features, drawn from the training scenes in our corpus, are used to train binary classifiers for each attribute. Fig. 5 gives example inputs and outputs.

We apply K-SVD as described in Sec. 4.2 to learn features for RGB-D images. Input data are now collections of $8\times8\times3$ image patches and $8\times8\times1$ depth patches, rather than four-dimensional distance vectors.

An object image is tiled into into cells, and the features of each cell E are the max pooled sparse codes, the component-wise maxima over all sparse codes within a cell:

$$f_I(E) = \left[\max_{j \in E} |x_{j1}|, \cdots, \max_{j \in E} |x_{jM}|\right] \quad (5)$$

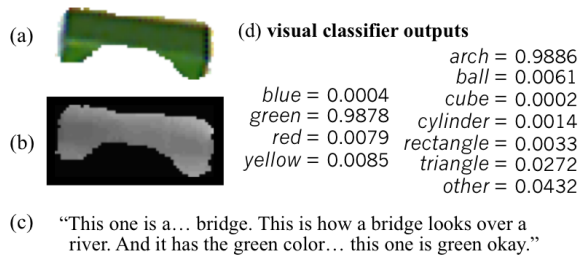

Figure 5: Data for one object in a scene. (a) and (b) show the RGB and depth signal from the camera; (c) is the spoken description. (d) gives the output of the visual shape and color classifiers.

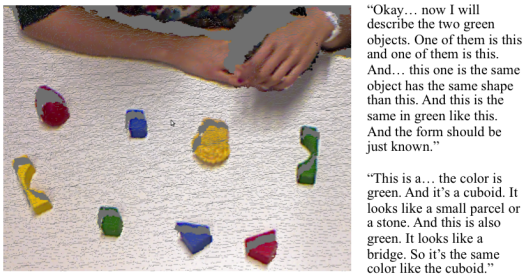

Figure 6: An image taken during data collection with examples of language used to describe the scene. We learn to connect words such as "cuboid" with appropriate classifiers.

Here, j ranges over all entries in the cell, and x_{jm} is the m-th component of the sparse code vector x_j of entry j. The feature f_I describing an image I is the concatenation of aggregated sparse codes in each spatial cell:

$$f_I = \left[F(E^1), \cdots, F(E^P)\right] \tag{6}$$

where E^p is a spatial cell generated by spatial partitions, and P is the total number of spatial cells.

Once these features are calculated, they are again used to train standard logistic regression classifiers for each attribute, which give probabilities for whether an object possesses each attribute. The outputs of individual classifiers are denoted $V \in \mathbb{R}$, where $v_{o,c}, c \in C$ and $v_{o,s}, s \in S$ are the outputs of appropriate color or shape classifiers for object o.

4.4 Language Model

At a high level, the goal of the language analysis step is to determine, for each object in a scene, whether the user is indicating that object. Intuitively, the likelihood r_o of an object o having been "referred to" depends on whether it has attributes the user mentions. The inputs for the language classifier are the language L, containing the user's description of the scene, plus the output of the attribute classifiers (for example, if L = "The blue one" and $v_{o_1,blue}$ has a high value, r_{o_1} should be high).

$$r_o = P(o|v_{i\in\{C,S\},o}, L) \tag{7}$$

In keeping with the goal of allowing free-form user input, the system does not use *a priori* information about what words correspond to what attributes; instead, such correspondences are learned from training data. This allows the system to learn interpretations of unexpected words, such as the use of "parcel" to describe cube-shaped blocks (Fig. 6). From the language L for all training scenes, we first extract a bag-of-words feature vector from W, the set of all words found in the corpus. (While this relatively simple language model introduces some errors, in general it works well for this data set, per the failure analysis in Sec. 5; in future we will extend it to a more complex model.) Each word $w \in W$ is a boolean feature $l_w \in \{1, 0\}$, whose value is its presence or absence in L; these features are analogous to the unordered codewords used for visual analysis. The scene description L can then be represented as a vector of these features. These word features are then combined with the output of the visual classifiers for each object (in practice, this is performed by training the language model using logistic regression with a polynomial kernel), to produce features for the pairwise co-occurrence of words and attributes:

$$\gamma_o = \left[l_{w1} \times v_{o,i1}, \ldots, l_{w|W|} \times v_{o,i|\{C,S\}|}\right], \quad w \in W, i \in \{C, S\} \tag{8}$$

Intuitively, this means that features encoding "good" correspondences will have high values when found with positive examples in the training data (e.g., $\langle l_{\text{"}bridge\text{"}} \times v_{o,arch}\rangle$ in Fig. 5), and can be weighted up accordingly.

We take the language L from each scene as a positive examplar for attributes of all positive objects. This introduces noise into the training data (for example, the first description in Fig. 6 might cause the classifier to learn a low value for the feature $\langle l_{\text{"}cuboid\text{"}} \times v_{o,cube}\rangle$, since there are no words describing shape), but avoids hand-labeling of the spoken language, in line with the long-term goal of having users train a robot without expert assistance.

4.5 Integration

Given a combination of language, visual attributes, and gesture, the object selection task is to automatically map a natural language scene description L, a (possible) gesture h, and a set of scene objects to the subset of objects G indicated by L and h, that is, $P(G \mid L, h)$. Rather than considering object subsets explicitly, we have factored this directly into individual classifications over objects.

A user may choose to use only gesture or only language; to avoid penalizing single-mode interactions, we use a small minimum value when the language or gesture signal is below some threshold ϵ, yielding the modified language and gesture scores of r'_o and h'_o. If there's no gesture present, ϵ may be interpreted as some kind of prior probability over whether an arbitrary object is being referred to or not. Experimentally, $\epsilon = 0.2$ worked well. The final score is then readily obtained by summing language and gesture to produce a final score k_o, the probability that object o is being indicated. This approach works well for our data; the fact that the additive combination of language and gesture causes the higher value to dominate is desirable. People can use language or gesture alone, so "not pointing at something" does not imply

excluding it from the indicated set. It is likely that other integration approaches will improve performance further, and it is an important direction for our future work.

5 Evaluation

Our corpus consists of data collected with 13 participants, each describing 28 scenes, yielding 364 language/video pairs. Testing was performed on a held-out set of 20% of these pairs, containing a total of 520 objects. Additional testing was performed by asking 5 new participants to each instruct the robot to perform tasks on 5 scenes (Sec. 5.5). In these trials, we use the participants' speech and gesture to label objects, and no additional training of the robot is performed.

5.1 Per-Object Accuracy

In this evaluation, objects are evaluated independently. To determine whether an object is being indicated, a cutoff is applied to the object score k_o, gesture score h'_o, or language+attribute score r_o; an object is positive (indicated) if its score is above the cutoff. This allows us to produce a precision/recall curve for the task (Fig. 7). As expected, the combination of gesture and language outperforms either.

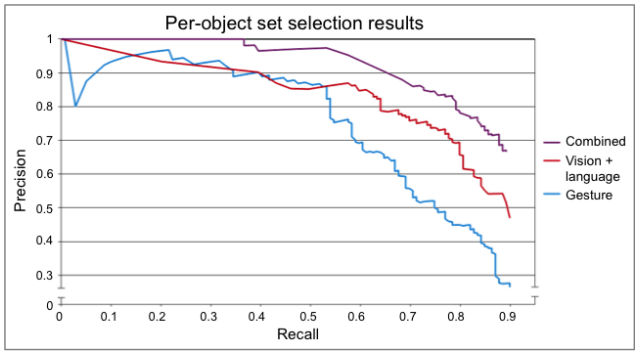

Figure 7: Precision (y-axis) and recall (x-axis), obtained from varying the cutoff above which object o is positive. The cutoff is applied to r'_o (for language+vision, red), h'_o (gesture only, blue), or k_o, which combines language and gesture (purple).

The system achieves a peak F1 score of 80.7%. As the cutoff increases, the system increasingly returns only correct objects, but misses more objects as well; depending on the context, different trade-offs of such false positives vs. false negatives might be appropriate.

5.2 Scene-Based Accuracy

In *scene-based* accuracy analysis, we again use object scores to determine whether objects are indicated, but treat a trial as a success only if all objects in the scene are correctly classified as positive or negative. This metric is strictly more difficult than the per-object analysis. At a naïve score cutoff of 0.5, our system performs perfectly on 53.9% of scenes; when the cutoff is tuned to best performance, which can be done using training data, the success rate rises to 57.9%. (For comparison, if the robot randomly selected some number of blocks, success would average 2.2%.)

5.3 Evaluation of Feasibility

Human evaluators were asked to perform the same task in order to test how well our corpus supports the task. Evaluators were asked to decide what objects are being indicated from a (silent) video, the transcribed language, or both. Each scene was shown to three evaluators; a successful trial is one in which *all* objects in a scene were identified correctly (scene-based). In cases of disagreement, we also calculate consensus majority-vote agreement (2 of 3) and test success of the consensus opinion. Results are shown in Table 1.

		Inter-annotator agreement	Success Rate
Individual	Vision+Language	0.799	0.866
	Gesture Only	0.780	0.803
	Given Both	0.747	0.873
Consensus	Vision+Language	0.961	0.888
	Gesture Only	0.964	0.830
	Given Both	0.967	0.926

Table 1: A trial is a "success" when *all* objects in a scene are labeled accurately. "Inter-annotator agreement" is percentage of trials where at least 2 testers agreed; in the "consensus" case, we test using these agreed-upon values. (top) Accuracy and inter-evaluator agreement across all evaluators; (bottom) accuracy of majority-voting results. When all three evaluators disagree, there is no consensus test.

5.4 Gesture Classification and Novel Features

We evaluated different approaches to determining which objects a person is gesturing to. As a baseline, we use simple closest-approach and closest leading-edge approach (as described in Sec. 4.1), and consider any object below a certain distance threshold to be "indicated". For a slightly less naïve baseline, we trained a logistic regression classifier over the minimum of the 4D leading-edge distances, computed over an entire interaction, for each object. This leading-edge distance performs better than simple distance-to-object, as when someone points to a block on the far edge of an arrangement, some part of the hand may be quite close to blocks nearer the speaker (see Fig. 9).

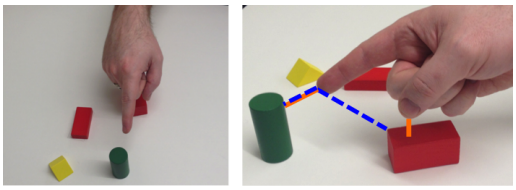

Figure 8: Orange lines show closest-approach distance to each object, while blue lines give leading-edge distance, demonstrating a common failure case; the green cylinder is the target.

We compare our novel hierarchical matching pursuit features against these baselines, using a logistic regression clas-

sifier. Fig. 9 shows precision/recall for all approaches.

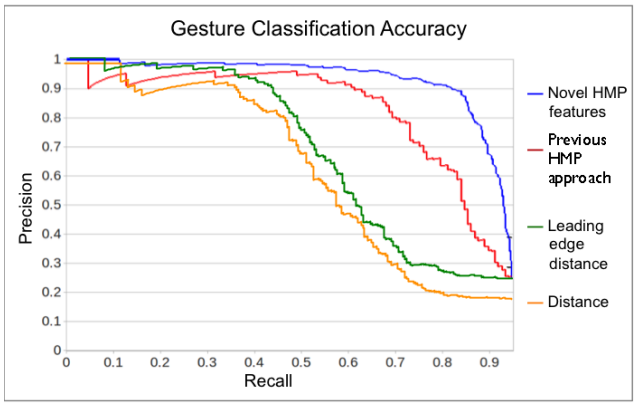

Figure 9: Precision (y-axis) and recall (x-axis) showing performance of different baselines. Yellow and green use the closest-approach distance and closest leading-edge distance, respectively. The red line shows the performance of a logistic regression classifier trained using traditional sparse codewords learned from the collection of four-dimensional vectors from the series, and the blue line uses our novel binary-value sparse codes.

Our high-dimensional hierarchical matching pursuit features provide significantly better results than the simple leading-edge vector. The difference is particularly striking in areas with high recall. For instance, our approach achieves 91% and 65% precision at 80% and 90% recall, whereas logistic regression over the closest approach only reaches 63% and 34% precision for the same recall values, respectively. This is due to the fact that people tend to move their hands in complex patterns, often moving closely over objects they do not want to indicate. While minimal distance is not able to capture such complex relationships, our HMP features are rich enough to provide good classification results.

5.5 Evaluation of Prototype System

We have implemented a prototype of a system on the Gambit manipulator platform (Matuszek et al. 2012a), using the models of language and gesture described above to identify and interact with objects being indicated by a user. Automatic speech recognition is performed using the Google Speech API. We conducted an initial, small evaluation of the system by asking five new participants to describe objects they would like the robot to put away in a bin. Even though many elements of the system (camera placement, specific objects, user) are different from the training data, the system does reasonably well. We report on the per-object accuracy of our results in Table 2.

The overall precision and recall of the combined system are 79% and 90%, overall consistent with results from testing our classification models. Results vary substantially by participant, reinforcing the belief that additional training by individual end users may be beneficial. In this trial, the combination of modalities is competitive, but not a clear improvement, which we believe to be a result of the weak language signal. The largest sources of errors in language interpretation were failures of voice recognition (46%), followed by overfitting to the training data resulting in poor shape classification (25%). The largest source of error in gesture (43%) was gestures made too far back from the robot, outside the expected workspace, which we will address by widening the robot's field of view and providing more transparency to the user regarding the robot's perceptions and state.

	Lang+attr			Gesture			Combined		
user	***P***	***R***	***F***	***P***	***R***	***F***	***P***	***R***	***F***
1	0.91	0.71	0.8	0.91	0.83	0.87	0.83	1.0	0.91
2	0.91	0.91	0.91	1.0	0.91	0.95	0.91	1.0	0.91
3	0.83	0.71	0.77	0.91	0.77	0.83	0.71	1.0	0.83
4	0.71	0.91	0.8	0.91	0.91	0.91	0.67	0.91	0.77
5	0.77	0.59	0.67	1.0	0.5	0.67	0.71	0.59	0.64
mean	0.83	0.77	0.8	0.95	0.75	0.84	0.79	0.9	0.84

Table 2: We report on performance for each participant across 5 scenes, containing 22 objects. Precision and recall are reported for each source of instruction and for the combined results.

6 Discussion & Future Work

We have described steps towards a robotic system that can learn, from examples, how users indicate objects using speech and gesture. Using a corpus of gesture and speech collected from untutored users, we train models that perform well, and we demonstrate that combining modalities of interaction results in more robust interpretation of human intent than either approach alone. We have constructed a prototype system which demonstrates the feasibility of applying the learned models to a system which performs simple tasks. In addition, we present results on several different approaches to identifying the target of unconstrained gesture, and demonstrate that our novel temporal HMP features can handle this complex problem.

In future, we intend to perform larger scale user studies on this system in order to quantify both how successfully the system can understand object indications and how well it can learn new concepts from on-the-fly interaction. Additionally, we intend to incorporate the more complex language model of (FitzGerald, Artzi, and Zettlemoyer 2013), which will improve our performance on understanding language, and the HMP-based object-recognition classifiers of (Sun, Bo, and Fox 2013), allowing us to extend our work to a richer set of objects in the world.

7 Acknowledgements

The work was funded in part by the Intel Science and Technology Center for Pervasive Computing (ISTC-PC), by ARO grant W911NF-12-1-0197, and through collaborative participation in the Robotics Consortium sponsored by the U.S. Army Research Laboratory under the CTA Program (Cooperative Agreement W911NF-10- 2-0016). We also thank Fortiss GmbH and Manuel Giuliani for work on gathering and annotating the data corpus, substantial assistance, and many helpful conversations.

References

Bo, L.; Ren, X.; and Fox, D. 2011. Hierarchical Matching Pursuit for image classification: architecture and fast algorithms. In *Advances in Neural Information Processing Systems*.

Clark, H. H. 2003. Pointing and placing. *Pointing: Where language, culture, and cognition meet* 243–268.

Clark, H. H. 2005. Coordinating with each other in a material world. *Discourse studies* 7(4-5):507–525.

Farhadi, A.; Endres, I.; Hoiem, D.; and Forsyth, D. 2009. Describing objects by their attributes. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, 1778–1785. IEEE.

FitzGerald, N.; Artzi, Y.; and Zettlemoyer, L. 2013. Learning distributions over logical forms for referring expression generation. In *Empirical Methods in Natural Language Processing (EMNLP)*.

Goodrich, M. A., and Schultz, A. C. 2007. Human-robot interaction: a survey. *Foundations and Trends in HCI* 1(3):203–275.

Lai, K.; Bo, L.; Ren, X.; and Fox, D. 2013. RGB-D object recognition: Features, algorithms, and a large scale benchmark. In Fossati, A.; Gall, J.; Grabner, H.; Ren, X.; and Konolige, K., eds., *Consumer Depth Cameras for Computer Vision: Research Topics and Applications*. Springer. 167–192.

Liu, P.; Glas, D. F.; Kanda, T.; Ishiguro, H.; and Hagita, N. 2013. It's not polite to point: generating socially-appropriate deictic behaviors towards people. In *Proceedings of the 8th ACM/IEEE international conference on Human-robot interaction*, 267–274. IEEE Press.

Mallat, S. G., and Zhang, Z. 1993. Matching pursuits with time-frequency dictionaries. *IEEE Transactions on Signal Processing* 41(12):3397–3415.

Matuszek, C.; FitzGerald, N.; Zettlemoyer, L.; Bo, L.; and Fox, D. 2012a. A joint model of language and perception for grounded attribute learning. In *Proc. of the 2012 Int'l Conference on Machine Learning*.

Matuszek, C.; Herbst, E.; Zettlemoyer, L.; and Fox, D. 2012b. Learning to parse natural language commands to a robot control system. In *Proc. of the 13th Int'l Symposium on Experimental Robotics (ISER)*.

McGregor, K. K.; Rohlfing, K. J.; Bean, A.; and Marschner, E. 2009. Gesture as a support for word learning: The case of under. *Journal of Child Language* 36.

Mitra, S., and Acharya, T. 2007. Gesture recognition: A survey. *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on* 37(3):311–324.

Mooney, R. 2008. Learning to connect language and perception. In Fox, D., and Gomes, C. P., eds., *Proc. of the Twenty-Third AAAI Conf. on Artificial Intelligence, AAAI 2008*, 1598–1601. Chicago, Illinois: AAAI Press.

Nickel, K., and Stiefelhagen, R. 2007. Visual recognition of pointing gestures for human–robot interaction. *Image and Vision Computing* 25(12):1875–1884.

Obaid, M.; Häring, M.; Kistler, F.; Bühling, R.; and André, E. 2012. User-defined body gestures for navigational control of a humanoid robot. In *Social Robotics*, volume 7621. Springer Berlin Heidelberg. 367–377.

Peltason, J.; Riether, N.; Wrede, B.; and Lütkebohle, I. 2012. Talking with robots about objects: a system-level evaluation in hri. In *Proceedings of the 7th annual ACM/IEEE international conference on Human-Robot Interaction*. ACM.

Perzanowski, D.; Schultz, A. C.; Adams, W.; Marsh, E.; and Bugajska, M. 2001. Building a multimodal human-robot interface. *Intelligent Systems, IEEE* 16(1):16–21.

Ramey, A.; González-Pacheco, V.; and Salichs, M. A. 2011. Integration of a low-cost rgb-d sensor in a social robot for gesture recognition. In *Human-Robot Interaction (HRI), 2011 6th ACM/IEEE International Conference on*, 229–230. IEEE.

Rouanet, P.; Danieau, F.; and Oudeyer, P.-Y. 2011. A robotic game to evaluate interfaces used to show and teach visual objects to a robot in real world condition. In *Human-Robot Interaction (HRI), 2011 6th ACM/IEEE International Conference on*, 313–320. IEEE.

Sandygulova, A.; Campbell, A. G.; Dragone, M.; and O'Hare, G. 2012. Immersive human-robot interaction. In *Proc. of the seventh annual ACM/IEEE international conference on Human-Robot Interaction*, HRI '12, 227–228. New York, NY, USA: ACM.

Steels, L., and Loetzsch, M. 2012. The grounded naming game. In *Experiments in Cultural Language Evolution*.

Steels, L. 2001. Language games for autonomous robots. *IEEE Intelligent systems* 16–22.

Suay, H., and Chernova, S. 2011. Humanoid robot control using depth camera. In *Human-Robot Interaction (HRI), 2011 6th ACM/IEEE International Conference on*, 401–401.

Sun, Y.; Bo, L.; and Fox, D. 2013. Attribute Based Object Identification. In *IEEE International Conference on on Robotics and Automation*.

Tellex, S.; Thaker, P.; Joseph, J.; and Rot, N. 2013. Learning perceptually grounded word meanings from unaligned parallel data. *Machine Learning Journal (Special Issue on Learning Semantics in Machine Learning)*.

Vollmer, A.-L.; Lohan, K. S.; Fischer, K.; Nagai, Y.; Pitsch, K.; Fritsch, J.; Rohlfing, K.; and Wrede, B. 2009. People modify their tutoring behavior in robot-directed interaction for action learning. Development and Learning, 2009. ICDL 2009. IEEE 8th International Conference on Development and Learning, 1–6. IEEE.

Wachs, J. P.; Kölsch, M.; Stern, H.; and Edan, Y. 2011. Vision-based hand-gesture applications. *Commun. ACM* 54(2):60–71.

Yang, J.; Yu, K.; Gong, Y.; and Huang, T. 2009. Linear spatial pyramid matching using sparse coding for image classification. In *Computer Vision and Pattern Recognition (CVPR)*. IEEE.

Yang, H.-D.; Park, A.-Y.; and Lee, S.-W. 2007. Gesture spotting and recognition for human–robot interaction. *Robotics, IEEE Transactions on* 23(2):256–270.