Toward Unconstrained Gesture and Language Interfaces

Cynthia Matuszek, Liefeng Bo, Luke Zettlemoyer, Dieter Fox
University of Washington Computer Science and Engineering
cynthia | lfb | lsz | fox @ cs.washington.edu

ABSTRACT

In this work, we investigate how people refer to objects in the world during relatively unstructured communication. We collect a corpus of object descriptions from non-expert users using language and naturalistic gesture to identify objects and attributes. This corpus is used to learn language and gesture models that enable our system to identify the objects referred to by a user. We demonstrate that combining multiple communication modalities is more effective for understanding user intent than focusing on only one type of input, and discuss the implications of these results on developing natural interfaces for interacting with physical agents.

Author Keywords

Vision, Natural Language, Language Grounding, Human-Robot Interaction

ACM Classification Keywords

H.5.2 Information interfaces and presentation (e.g., HCI): Miscellaneous.

General Terms

Algorithms, Human Factors

INTRODUCTION

As computing systems move into the real world, the importance of enabling untrained users to interact with them in a natural way increases. For instance, interactive table top systems [40] or mobile robots operating in populated environments [5] require interfaces that go beyond touch or keyboard. Fortunately, the physical capabilities of such agents—which can potentially see, hear, speak, and so on—offer new possibilities for the development of natural user interfaces that take advantage of the different ways that humans interact with the physical world. Natural language and gesture are rich, intuitive mechanisms for people to describe objects, issue instructions, and convey intentions [11].

Existing multimodal interfaces, however, require the user to learn how a system expects to be instructed, rather than the system learning how the user naturally communicates. Gestural interfaces have primarily focused on gesture *recognition*—that is, focused on identifying a lexicon of gestures which the user must learn [14]. Natural language, which can encode substantial complexity, is still an incomplete communication mechanism in a physical setting, where it is often natural to use gesture to focus attention [34]. Compare, for example, "Put the mug that's on the table in the cupboard two to the left of the stove," versus "Put this mug in there." The latter is a natural way of communicating a need

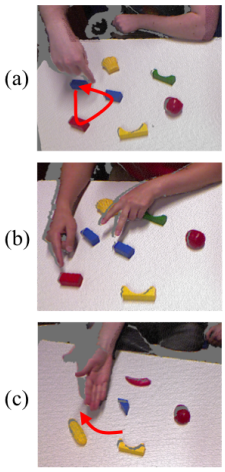


Figure 1: Examples of unscripted gesture and language investigated in this paper. (a) A circular pointing motion looping around the objects; (b) pointing with multiple fingers and both hands; (c) an open-handed sweep above objects. (d) and (e) give examples of different language for the two scenarios shown (a/b and c). The grammatical statements and errors are typical of spoken language.

to a robot, set in a context where traditional input devices such as keyboards are lacking. As a result, there is long-term interest in interfaces that incorporate multiple modes of interaction, particularly gesture and language [4, 8], and especially for human-robot interactions [23].

Unfortunately, human language and gesture can be quite complex, as shown in the examples in Fig. 1. Someone referring to a set of blocks as the "blue red *(sic)* rectangles," while sketching a rough circle above them, is entirely comprehensible to a person, but outside the scope of current user interfaces—a situation which will become harder to accept as robots and interactive systems become more widely deployed and capable.

Our goal in this work is to demonstrate steps towards interfaces where an individual can, for instance, teach a robot to understand combinations of the unconstrained ways a person might wish to communicate. We introduce a system that learns to interpret user intent in a physical workspace from unstructured communication. Using low-cost RGB-D (RGB + depth) cameras, we collected a corpus of interactions from people tasked with identifying objects in a space, and used those to train language, gesture, and vision models. Those models were then applied to the task of selecting the objects in a physical workspace a person is trying to indicate. Combining these modalities performs better at captur-

ing user intent than using unconstrained language or gesture in isolation, without asking the user to become familiar with a particular lexicon of gestures or commands.

Our contributions are as follows.

1. A large dataset with 364 annotated videos showing people describing objects in a table top setting.
2. A technique for unsupervised learning of features that are rich enough to enable accurate recognition of complex human gestures.
3. A recognition model that combines language, gestures, and visual object attributes.
4. A technique for learning the joint recognition model from weakly annotated training data.

While the specific task we target in this paper uses a limited set of objects and attributes, understanding how people naturally refer to objects in the world is common to a large set of scenarios. Being able to understand natural human communication has the potential to enable the development of new interfaces in areas as diverse as entertainment [29], medical technology [37], search-and-rescue [27], human-robot interaction [10], and assistive care [12].

RELATED WORK

The development of natural interfaces that allow humans to interact with technology in the physical world is a problem of general importance [1, 38]. A number of different modalities have been explored to support such interfaces [36]. This work builds on natural language understanding, gestural input, and human-robot interaction particularly.

Gesture has been extensively explored for use in such interfaces [22, 19]. Our work is most closely related to work on using gesture to control physical systems, such as a quadcopter or humanoid robot [30]. However, rather than performing recognition of a predefined gestures (arguably not optimal for natural user interfaces [20]), our system is trained on a corpus of examples of unscripted gestures.

Natural language understanding for the development of usable interfaces is currently a substantial research area in HRI (Human-Robot Interaction), of particular interest to pervasive computing [25]. The overlap between natural language and physical agents has led to intense study of the field of *natural language grounding*, or interpreting language in the context of physical sensing and actuation. Learning about the world and human communication from a combination of language and sensor data has achieved particular success in understanding commands, for example in navigation [9, 17] robot command interpretation [33], and search and rescue [6]. Although in this work we assume the set of colors and shapes is known in advance, other work in the area [16] has demonstrated the capability of a similar system to learn visual classifiers for previously-unseen attributes.

The proliferation of consumer-grade depth cameras such as the Microsoft Kinect has had a substantial impact on the feasibility of developing come-as-you-are interfaces without user-based equipment such as accelerometers [36]. The usefulness of these devices has been demonstrated in tracking user movement and hand gestures, [21] detecting and responding to human information about objects in the actions [35], and gathering information about objects in the world [31].

The usefulness of these devices has been enhanced by the existence of standard toolkits for handling the point clouds they produce; in this work we use the PCL library [26] for vision and ROS, the Robot Operating System [24], to manage communication and visualization. We similarly take advantage of work in the computer vision community on the study of features suitable for object identification [3].

Combining different input modalities for natural user interfaces is a tremendously important area [18, 7, 8], and has already been used with some success in controlling the navigation of a robotic platform [32]. However, to the best of our knowledge, previous work on combining modalities in natural user interfaces does not learn from examples to handle unconstrained user input. Our goal of understanding attribute descriptions is most similar to work on using physical sensor data to learn about object attributes [16], but our work is on enabling natural interfaces rather than on collecting new linguistic concepts.

PROBLEM STATEMENT AND DATA COLLECTION

The ultimate goal of our work is to learn to understand the intentions and references of users, from the users themselves, without requiring specialized training. To build our system, we first recorded examples of people describing objects after being given minimal instructions. That data set was used to train models of different interaction styles, which could then be used to tackle the object selection problem.

The Object Selection Task

Our task is to build a system that understands someone who is indicating objects found in a physical workspace, by verbally describing any combination of object attributes, by gesturing, or both. For each scenario—consisting of an RGB-D video of objects and people's interactions with them, plus transcribed language—we want to identify the objects in the scene that are indicated (or positive).

Because the goal is to robustly understand unconstrained user input, the indications do not need to follow a specific format (e.g., a list of allowable gestures), and the system tries to identify objects even when user input is partial, redundant, or even insufficient.

Data Collection and Corpus

In order to collect information about how people refer to objects when given few constraints, we used the Kinect sensor (usually mounted on a robotic manipulation platform) to record people describing objects with language and gesture.

Participants were instructed to distinguish certain objects from a collection of objects, using language and gesture "as if they

	total scene	objects / scene (avg)	positive obj./ scene (avg)	pluralism		arrangement		participants		description (longest)	description (shortest)	description length (avg)
				1 positive	*2+ positives*	*spread*	*clustered*	*(m)*	*(f)*			
corpus	364	6.89	2.64	130	234	117	247	7	6	111 s	2.8 s	18.65 s

Table 1: **Data in the collected corpus. 13 participants were asked to describe each of 28 scenes, giving a total of 364 language/gesture/vision data tuples. Objects in scenes were arranged to vary whether positive objects were clustered together or spread out, and whether there were one or more positive objects in the set. Participants were given minimal instruction on how to describe scenes, leading to a wide spread of language complexity and time spent per scene.**

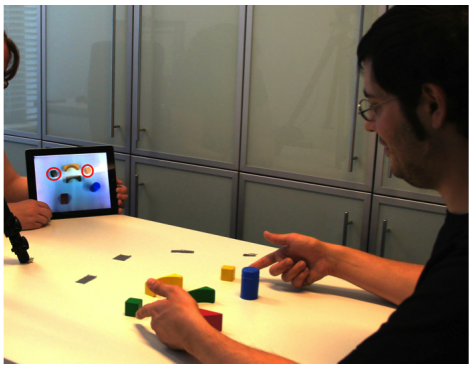


Figure 2: **A data collection participant indicating objects. The experimenter shows an iPad with target objects circled in order to avoid providing linguistic or gestural cues that might alter participant behavior.**

were describing those objects to a robot." Participants were not given a predefined set of gestures or instructions to use, and experimenters provided prompts by showing images of the table layout with objects circled, in order to avoid providing linguistic or gestural examples (see Fig. 2).

Twenty-eight different arrangements of objects were used, varying across dimensions such as whether the objects of interest were clustered together or separated, how many of the objects on the table were to be indicated, and what attributes (color and shape) the objects did or did not have in common. On average, scenes contained seven blocks, of which an average of two are to be indicated. Thirteen participants described each scene.

The length of responses from participants ranged from very brief (around three seconds) to more than a minute, with a median instruction time of roughly 18.5 seconds. The resulting data set contains examples of language used without gesture, gesture paired with non-descriptive language (e.g., "These three objects"), and gesture and language used together. In most cases, participants provided redundant information, where either gesture or language would be sufficient to convey intent. Statistics are given in Table 1.

Evaluation of Feasibility

Because we are using only the subset of the available data that can be detected by our sensors (the RGB-D camera and microphone), we are interested in learning how well that data supports the object selection task overall. We used Mechanical Turk to present human evaluators with the same object selection task we ask the system to perform.

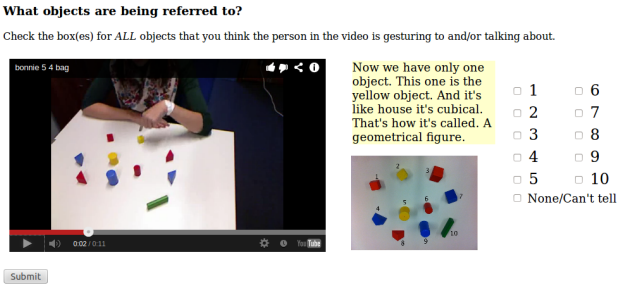


Figure 3: **The task presented to our Mechanical Turk workers. Workers were asked what objects are being indicated, given video, language, or both (shown).**

Evaluators are presented with a (silent) video of a person gesturing to objects, the text of a person talking about those objects, or both, and asked to indicate the objects being indicated (see Fig. 3). Each task was given to three evaluators.

We report results on the complete collected data, but also on consensus accuracy, in which a majority of our evaluators agreed on an interpretation (a standard technique for improving the quality of crowdsourcing results [28]). Results are shown in Table 2.

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

Table 2: **Inter-annotator agreement and error rate, measured per-scene. (top) Accuracy and inter-evaluator agreement across all evaluators; (bottom) for majority-voting results.**

For majority-vote, inter-evaluator agreement is consistently around 96%. And evaluation achieves an 83%–93% success when evaluated on whether all events were correctly identified (reported as *per-scene accuracy* in the Evaluation section). Consistent with our hopes for multimodal learning, both the system and human evaluators perform better when given multimodal data.

SYSTEM OVERVIEW

Fig. 4 shows the overall architecture of the object identification interface, starting with human input. First, RGB-D video data and language are collected from someone trying to indicate objects in a workspace. Objects and hands are identified, and features are extracted that provide informa-

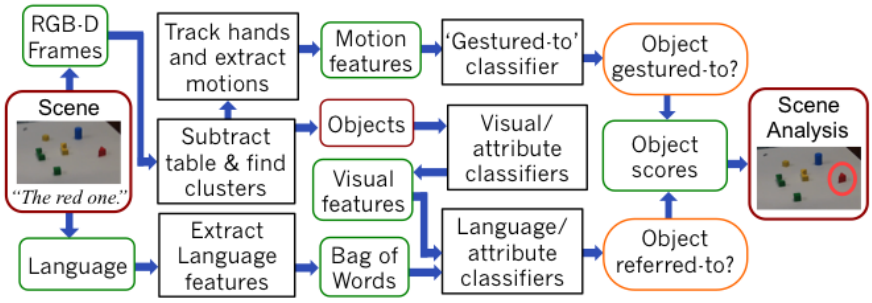


Figure 4: **An overview of the system. Video frames are used to locate objects and track hand movements, and language is recorded. Motion features are used to classify which objects are being gestured to, and classifiers for object attributes (such as color) are combined with language to determine whether an object is referred to. These are combined to determine what object(s) are being indicated.**

tion about the color and shape of objects, gestures made, and language used. From those features, a set of classifiers determines whether an object is being gestured to or spoken about; finally, those outputs are combined to determine the set of objects being referred to.

To accomplish the goal of identifying objects from whatever interactions a user offers, we first train individual components of the system on example interactions. The system contains the following components: (1) a classifier that interprets gestures that indicate specific referents; (2) visual classifiers that correspond to the appropriate object properties; (3) a models of how words correlate to the visual attribute classifiers. The system then parses new user interactions via the following pipeline.

First, raw sensor data is tranformed into a form that has meaning at the semantic or conceptual level. RGB-D points are processed in order to automatically separate and identify hands, objects, and the shared agent/human workspace. Similarly, speech is transcribed, although ultimately speech recognition would be an integral part of a user interface.

Once objects and hands are identified, the system extracts state of the art visual hierarchical matching pursuit (HMP) features [3] from RGB and depth data, followed by logistic regression to classify each object's color and shape. If a participant chooses to use gesture, the system identifies point clusters that represent one or more moving hands in the working area. Geometric features are extracted from depth values at every frame, and after training, logistic regression is used to determine, for every object, whether the hand gestures to that object during the video.

Any language from the user is analyzed at a per-scene level to determine whether it makes references to object attributes. If so, we consider objects that have those attributes to be 'referred to,' and therefore a likely target of the user's intention. We restrict language and vision classification to a known set of color and shape attributes and do not explicitly address other forms of description, instead concentrating on features that perform well for the object selection task. Bayesian logistic regression is used to determine whether a scene's language refers to particular attributes.

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.

APPROACH

In order to understand a user's references to the world, his/her input goes through a number of steps. In this section, we go into detail on each phase of processing a scene:

- Raw sensor input is processed to identify the workspace, objects, and hands; language is transcribed.
- If hands are present, a gesture classifier is used to obtain a probability that each object was gestured to.
- Vision classifiers return probabilities for the color and shape of each object.
- Language is combined with the output of the vision classifiers to determine whether each object is referred to in speech.
- 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 a final evaluation of user intent.

Point-Cloud Processing

In this first step, our system extracts the user's hands and the objects from each frame of the Kinect video. An individual Kinect frame provides a set of points, or *point cloud*, each of which has an associated RGB value and a depth corresponding to its distance from the camera. First, everything but the known workspace is cropped, and a fixed transformation is applied to rotate the point cloud to be axis-aligned to the edge of the table. (Because the Kinect is mounted above and to one side of the human interactor, this results in a significant viewpoint change; see Fig. 5.)

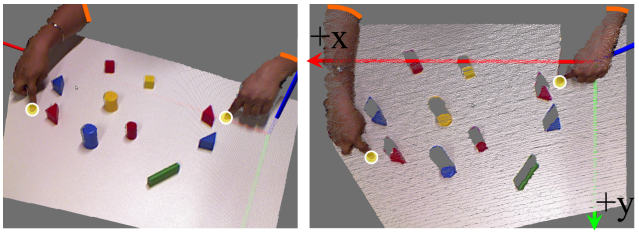


Figure 5: **A frame from the data corpus, cropped to the workspace (left), then *x,y* axis-aligned to the table (right). Moving clusters of points that extend beyond the workspace are hands (boundaries in orange), while static clusters on the table are objects. Areas where no 3D points exist, due to occlusion or resolution, are gray. The leading-edge point on each hand (yellow) is used to calculate gesture features.**

To automatically segment individual objects $o \in O$ from each scene, RANSAC plane fitting is performed on point depth values to remove the table plane, then connected components (segments) of points more than 0.5 cm above that plane are extracted. Remaining points are segmented into clusters of points that are within 2 cm of one another. Fig. 6 shows an example of a segmented object. Because our setting involves people describing static objects, objects can be treated as persistent. Small, non-moving clusters on the table are registered as objects, while larger, moving clusters that extend past the working area of the table are hands.

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 as the point on the hand that is furthest from the user's body, determined by the highest value in the y-axis (see yellow markers in Fig. 5). We then compute the distances between this point and the centroid of each object. 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.

In practice, the system uses the first five frames of the video, in which no significant motion generally occurs, to locate objects, then stores the resulting point cloud for each object rather than re-analyzing objects every frame. Point clouds representing hands are always re-evaluated.

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 some way. Unfortunately, this task turns out to be quite complex, due to the substantial variability in how people use their hands to refer to objects (see Fig. 1). Our first approach was to learn a classifier based on the distances between an object and the user's hand during the interaction. While such an approach provides reasonable results, a richer set of features can provide substantial gain in classification performance, as we show in our experimental evaluation. To learn rich features, we use a technique called sparse coding, which has become a popular tool in several fields, including signal processing and object recognition [39].

As described in the Point-Cloud Processing section, we extract a sequence of four-dimensional vectors of spatial features from the video. Rather than applying a classifier directly to this sequence, we apply sparse coding to learn features that are more suitable for the classification task. Sparse coding models data as sparse linear combinations of codewords selected from a codebook [13]. 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.

In our gesture recognition task, the data are the collection of four dimensional vectors from the entire time-series T, the sequence of frames in which a person was interacting. Here, we use efficient 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:

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

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, the zero-norm $\|\cdot\|_0$ counts non-zero entries in the sparse codes x_n, and 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 [3]. 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:

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

We design a K-SVD-like algorithm to decompose the above optimization problem into two subproblems: ENCODING and CODEBOOK UPDATE, which are solved in an alternating manner. At each iteration, the current codebook D is used to encode the data Y by computing the sparse code matrix X by matching pursuit [15] (ENCODING). Then, the codewords of the codebook are updated one at a time by gradient descent optimization, resulting in a new codebook (UPDATE). The new codebook is used in the next iteration to recompute the sparse code matrix followed by another round of codebook update, repeated until the maximum iteration is reached.

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 maximize each component of binary sparse codes of four

dimensional vectors over all timesteps, and generate 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 use logistic regression to train a classifier, which gives h_o, the probability that object o is being gestured to with one or both hands.

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

where Θ_g^P is the parameters in Θ^P for the gesture classifier. This combination of features outperforms the naïve closest approach distance, as discussed in Evaluation.

Color and Shape Classification

The goal of this step is to determine what attributes each object in a scene has—that is, to find its color and shape. 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:

$C = \{blue, green, red, yellow\}$
$C = \{arch, ball, cube, cylinder, rectangle, triangle, other\}$

Once segmented objects are obtained, we extract the same hierarchical matching pursuit (HMP) features as are used for gesture. These features have been successfully used for object attribute analysis in similar contexts [31]. These features, drawn from the training scenes in our corpus, are used to train binary classifiers for each attribute in C and S. These classifiers are then run on on each object o. See Fig. 6 for example color and depth image inputs of an object, and outputs for classifiers.

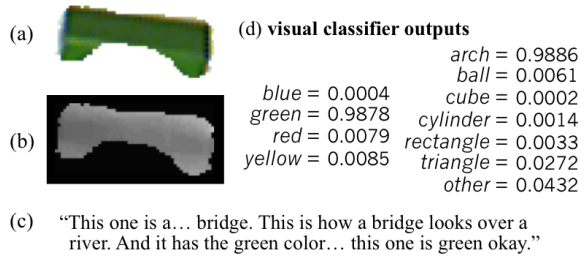


Figure 6: **Data for one object in a scene, after automatic segmentation. (a) and (b) show the RGB and depth signal from the camera; (c) is a transcription of the spoken description. (d) gives the output of the visual shape and color classifiers.**

We apply the binary K-SVD features described in Gesture Classification to learn features for RGB-D images. Input data are now collections of $8 \times 8 \times 3$ image patches and $8 \times 8 \times 1$ depth patches, rather than four-dimensional distance vectors. Spatial max pooling instead of temporal max pooling is applied with the learned codebooks to aggregate the sparse codes.

A cropped object image I is divided spatially into smaller cells. The features of each spatial cell E are the max pooled sparse codes, which are simply the component-wise maxima over all sparse codes within a cell:

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

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 are the concatenation of aggregated sparse codes in each spatial cell

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

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 possible shape and color attribute, which then output the probability that any new object encountered 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.

Language Model

At a high level, the goal of the language analysis step is the same as that of gesture analysis: for each object in a scene, determine whether the user is indicating that object. Intuitively, an object o has been "referred to," expressed as r_o, if it has attributes the user mentions. Accordingly, the inputs for the language classifier are the transcribed speech, L, containing the user's description of the scene, plus the output of the vision classifiers.

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

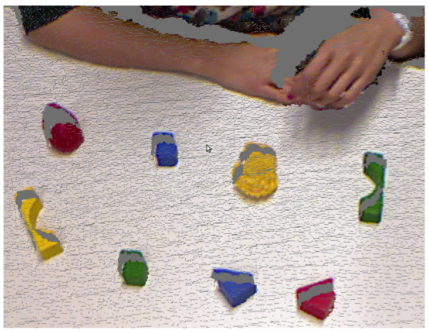
For example, if L = "The blue one" and $v_{o_1,blue}$ has a high value, r_{o_1} should be high.

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 correct interpretations of unexpected words, such as the recurring use of "parcel" and "package" to describe cube-shaped blocks, as in Fig. 7.

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. Each individual word $w \in W$ is as 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 pairwise 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 co-occurrence of words and attributes:

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



"Okay… now I will describe the two green objects. One of them is this and one of them is this. And… this one is the same object has the same shape than this. And this is the same in green like this. And the form should be just known."

"This is a… the color is green. And it's a cuboid. It looks like a small parcel or a stone. And this is also green. It looks like a bridge. So it's the same color like the cuboid."

Figure 7: **An image taken during data collection, and examples of language used to describe the scene, showing errors typical of spoken language. Given knowledge of objects' attributes from vision, our goal is to connect words such as "green" and "cuboid" with objects displaying those characteristics.**

Intuitively, this means that features encoding "good" correspondences will have high values when found with positive examples in the training data (e.g., $\langle l_{\text{"bridge"}} \times v_{o,arch} \rangle$ in Fig. 6), 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. 7 might cause the classifier to learn an inappropriately low value for the feature $\langle l_{\text{"cuboid"}} \times v_{o,cube} \rangle$, 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.

Integration of Vision, Language & Gesture

Since users are free to use any combination of language and gesture when indicating objects, the final step is to combine the output of all the classifiers to produce a final *object score* that represents whether our system believes the user is indicating that object.

More formally, given a combination of language, vision, 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 classifiers over objects.

Because we are combining possibly-redundant ways of referring to objects, a user may choose to use only gesture or only language. Since we do not wish to penalize single-mode interactions, we use a small minimum value when the language or gesture signal is below some threshold ϵ:

$$r'_o = \begin{cases} r_o, & r_o \geq \epsilon \\ \epsilon, & r_o < \epsilon \end{cases} \qquad h'_o = \begin{cases} h_o, & h_o \geq \epsilon \\ \epsilon, & h_o < \epsilon \end{cases} \tag{9}$$

Experimentally, we found that $\epsilon = 0.2$ worked well for the data set; however, our results were not particularly sensitive to small changes in ϵ.

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:

$$k_o = (r'_o + h'_o) \tag{10}$$

To determine whether an object is being indicated using language or gesture in isolation (as in Fig. 10), r'_o or h'_o can be used directly.

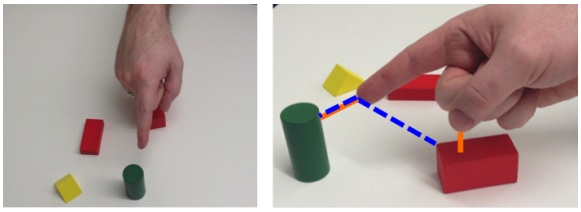


Figure 8: **Leading-edge versus closest distance: (left) From the perspective of the physical agent; (right) side view. Orange lines show the closest-approach distance between the hand and the two front objects, while blue lines show the leading-edge distance. The actual shortest distance is to the red block, while the shortest leading-edge approach is consistent with the person's intent.**

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.

Gesture Recognition

In order to investigate the complexity and usefulness of gesture recognition, we evaluated different approaches to determining which objects a person is gesturing at. As baseline, we trained a logistic regression classifier over the minimum of the 4D leading-edge distances between the object and the person's hand, computed over an entire interaction (see Point-Cloud Processing). In our early tests, we found the leading-edge distance to provide significantly better results than the minimum distance between any point on the hand and the object. One explanation for this is that the participant is communicating with the interlocutor to whom they have been asked to describe objects, and position their hands to make sense from that perspective (see Fig. 8).

We then trained a logistic regression classifier using our novel hierarchical matching pursuit features described in the Gesture Classification section. Fig. 9 shows precision / recall curves for logistic regression on minimum leading-edge and on our features. As can be seen, 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 65% and 91% precision at 90% and 80% recall, whereas the baseline approach only reaches 34% and 63% precision for the same recall values, respectively. This is due to the fact that people do not clearly point at individual objects, but they tend to move their hands in much more complex patterns, often moving closely over objects they do not want to indicate. While a minimial leading-edge value is not able to capture such complex relationships,

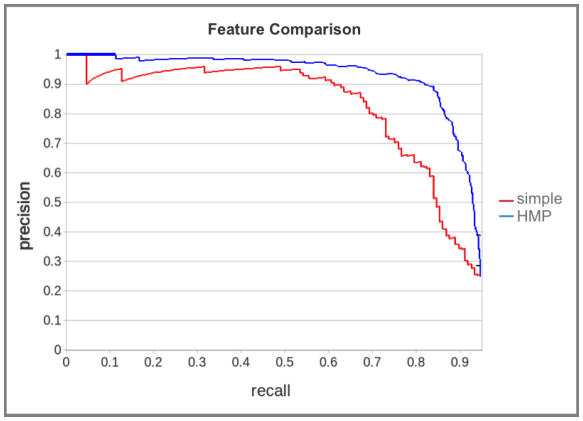


Figure 9: **Precision (y-axis) and recall (x-axis), obtained from varying the cutoff above which object o is considered to have been indicated (positive). The red line shows the performance of a classifier trained directly over the 4D feature vectors extracted from the video, and the blue line over higher-level hierarchical matching pursuit-derived features.**

our hierarchical matching pursuit features are rich enough to provide good classification results.

Per-Object Accuracy

In this evaluation, objects are evaluated independently, making it possible to get some objects in a particular scene right and others wrong. To determine whether an object is being indicated, a cutoff is applied to the object score k_o, gesture score h'_o, or language+vision score r_o; the system considers an object positive (indicated) if its score is above the cutoff. This allows us to vary the cutoff to produce a precision/recall curve for the task (Fig. 10). As expected, the combination of interaction modes outperforms either gesture or language alone.

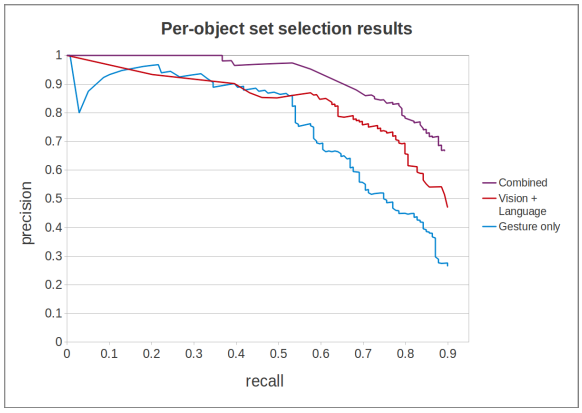


Figure 10: **Precision (y-axis) and recall (x-axis), obtained from varying the cutoff above which object o is positive. The cutoff is applied to r_o (for language+vision, red), h'_o (gesture only, blue), or k_o, which combines language and gesture (purple). Combining input data produces better results than using only one source of input.**

The system achieves a peak F1 score (a weighted average of precision and recall) 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.

Note that at a recall of 90%, gesture only achieves a precision of 26%, the combination of visual classifiers and language reaches 47% precision, and the combination of all information sources increases this precision to 66%, indicating that combining all sources of information can drastically increase the capabilities of systems requiring interactive object identification.

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%.

Discussion

It seems clear that combining modalities of interaction—here, gesture combined with relatively unconstrained natural language—resulted in more robust interpretation of human intent than using only one approach. This conclusion is consistent with the observation that combining gesture and language reduces ambiguity in user interfaces [18].

Comparing our results to those obtained by human evaluators (see Evaluation of Feasibility) is is informative. Unlike our system, human interpretation of language only outperforms gesture only by a nontrivial margin, which strongly suggests that our simple mechanism for language could be improved by using a more complex language model, such as those found in work by Artzi & Zettlemoyer [2].

DISCUSSION & FUTURE WORK

As robots and physical sensors become more ubiquitous, the value of interfaces that allow people to interact with them naturally and comfortably increases as well. In this work, we have described steps towards an interface which improves on that goal in two ways. First, our system takes language, gesture, and perception of a scene and uses them to performed joint, unsupervised learning of how to understand a human's intentions; this joint reasoning improves on using only one kind of interaction. Second, our system learns from examples how the people in our data set communicate naturally, without making making strong suppositions about what modes they are most comfortable using, and without requiring them to learn how to interact with the system. Pushing the learning from the human to the system in this way improves on some of the ways in which current interfaces are less than natural.

We train our system and evaluate it on a corpus of video, depth information, and transcribed language of people interacting with objects in whatever way they find comfortable. Its performance strengthens our belief that this sort of unconstrained, natural interaction can be used to build successful interfaces. An evaluation of human performance on the same

corpus suggests that it is, in fact, an appropriate resource for further exploring and improving on the task of understanding how people refer to objects.

Understanding references to things in the word is already a necessary component of many tasks; however, we believe that our approach has the potential to scale to similarly learning how humans interface with systems in more elaborate contexts. In future, it is our intention to improve the system presented here, and to apply this kind of interaction learning to other areas.

ACKNOWLEDGEMENTS

(Sponsorship acknowledgement redacted for blind review)

REFERENCES

1. G. D. Abowd and E. D. Mynatt. Charting past, present, and future research in ubiquitous computing. *ACM Transactions on Computer-Human Interaction (TOCHI)*, 7(1):29–58, 2000.
2. Y. Artzi and L. Zettlemoyer. Bootstrapping semantic parsers from conversations. In *Proc. of the Conference on Empirical Methods in Natural Language Processing*, pages 421–432. Association for Computational Linguistics, 2011.
3. L. Bo, X. Ren, and D. Fox. Hierarchical Matching Pursuit for image classification: architecture and fast algorithms. In *Advances in Neural Information Processing Systems*, December 2011.
4. R. A. Bolt. *"Put-that-there": Voice and gesture at the graphics interface*, volume 14. ACM, 1980.
5. W. Burgard, A. B. Cremers, D. Fox, D. Hähnel, G. Lakemeyer, D. Schulz, W. Steiner, and S. Thrun. Experiences with an interactive museum tour-guide robot. *Artificial Intelligence*, 114(1-2):3–55, 1999.
6. R. Cantrell, K. Talamadupula, P. Schermerhorn, J. Benton, S. Kambhampati, and M. Scheutz. Tell me when and why to do it!: Run-time planner model updates via natural language instruction. In *Proc. of the 2012 Human-Robot Interaction Conference*, Boston, MA, March 2012.
7. A. Cheyer and L. Julia. Multimodal maps: An agent-based approach. *Multimodal Human-Computer Communication*, pages 111–121, 1998.
8. J. Jain, A. Lund, and D. Wixon. The future of natural user interfaces. In *Proc. of the 2011 annual conference extended abstracts on Human factors in computing systems*, pages 211–214. ACM, 2011.
9. J. Kim and R. J. Mooney. Unsupervised PCFG induction for grounded language learning with highly ambiguous supervision. In *Proc. of the Conference on Empirical Methods in Natural Language Processing and Natural Language Learning (EMNLP-CoNLL '12)*, pages 433–444, Jeju Island, Korea, July 2012.
10. D. Kortenkamp, E. Huber, R. P. Bonasso, et al. Recognizing and interpreting gestures on a mobile robot. In *Proc. of the National Conference on Artificial Intelligence*, pages 915–921, 1996.
11. M. W. Krueger. *Artificial reality II*, volume 10. Addison-Wesley Reading (Ma), 1991.
12. V. A. Kulyukin. On natural language dialogue with assistive robots. In *Proc. of the 1st ACM SIGCHI/SIGART conference on Human-robot interaction*, pages 164–171. ACM, 2006.
13. K. Lai, L. Bo, X. Ren, and D. Fox. RGB-D object recognition: Features, algorithms, and a large scale benchmark. In A. Fossati, J. Gall, H. Grabner, X. Ren, and K. Konolige, editors, *Consumer Depth Cameras for Computer Vision: Research Topics and Applications*, pages 167–192. Springer, 2013.
14. A. Malizia and A. Bellucci. The artificiality of natural user interfaces. *Communications of the ACM*, 55(3):36–38, 2012.
15. S. G. Mallat and Z. Zhang. Matching pursuits with time-frequency dictionaries. *IEEE Transactions on Signal Processing*, 41(12):3397–3415, 1993.
16. C. Matuszek*, N. FitzGerald*, L. Zettlemoyer, L. Bo, and D. Fox. A joint model of language and perception for grounded attribute learning. In *Proc. of the 2012 Int'l Conference on Machine Learning*, Edinburgh, Scotland, June 2012.
17. C. Matuszek, E. Herbst, L. Zettlemoyer, and D. Fox. Learning to parse natural language commands to a robot control system. In *Proc. of the 13th Int'l Symposium on Experimental Robotics (ISER)*, 2012.
18. C. Mignot, C. Valot, and N. Carbonell. An experimental study of future "natural" multimodal human-computer interaction. In *INTERACT'93 and CHI'93 conference companion on Human factors in computing systems*, pages 67–68. ACM, 1993.
19. S. Mitra and T. Acharya. Gesture recognition: A survey. *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, 37(3):311–324, 2007.
20. D. A. Norman. Natural user interfaces are not natural. *Interactions*, 17(3):6–10, 2010.
21. I. Oikonomidis, N. Kyriazis, and A. Argyros. Efficient model-based 3d tracking of hand articulations using kinect. *BMVC, Aug*, 2, 2011.
22. V. I. Pavlovic, R. Sharma, and T. S. Huang. Visual interpretation of hand gestures for human-computer interaction: A review. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19:677–695, 1997.
23. D. Perzanowski, A. C. Schultz, W. Adams, E. Marsh, and M. Bugajska. Building a multimodal human-robot interface. *Intelligent Systems, IEEE*, 16(1):16–21, 2001.

24. M. Quigley, K. Conley, B. P. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A. Y. Ng. Ros: an open-source robot operating system. In *ICRA Workshop on Open Source Software*, 2009.

25. C. Ramos, J. C. Augusto, and D. Shapiro. Ambient intelligence - the next step for artificial intelligence. *Intelligent Systems, IEEE*, 23(2):15–18, March-April.

26. R. B. Rusu and S. Cousins. 3D is here: Point cloud library (pcl). In *International Conference on Robotics and Automation*, Shanghai, China, 2011 2011.

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

28. R. Snow, B. O'Connor, D. Jurafsky, and A. Y. Ng. Cheap and fast—but is it good?: evaluating non-expert annotations for natural language tasks. In *Proc. of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 254–263. Association for Computational Linguistics, 2008.

29. T. Starner, B. Leibe, B. Singletary, and J. Pair. Mind-warping: towards creating a compelling collaborative augmented reality game. In *Proc. of the 5th international conference on Intelligent user interfaces*, pages 256–259. ACM, 2000.

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

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

32. G. Taylor, R. Frederiksen, J. Crossman, M. Quist, and P. Theisen. A multi-modal intelligent user interface for supervisory control of unmanned platforms. In *Int'l Conference on Collaboration Technologies and Systems (CTS)*, pages 117–124. IEEE, 2012.

33. M. Tenorth and M. Beetz. A unified representation for reasoning about robot actions, processes, and their effects on objects. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Vilamoura, Portugal, October, 7–12 2012.

34. E. A. Topp, H. Huettenrauch, H. I. Christensen, and K. S. Eklundh. Bringing together human and robotic environment representations-a pilot study. In *Intelligent Robots and Systems, 2006 IEEE/RSJ International Conference on*, pages 4946–4952. IEEE, 2006.

35. N. Vidakis, M. Syntychakis, G. Triantafyllidis, and D. Akoumianakis. Multimodal natural user interaction for multiple applications: The gesturevoice example. In *Telecommunications and Multimedia (TEMU), 2012 International Conference on*, pages 208–213. IEEE, 2012.

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

37. J. P. Wachs, H. I. Stern, Y. Edan, M. Gillam, J. Handler, C. Feied, and M. Smith. A hand gesture sterile tool for browsing MRI images in the OR. *Journal of the American Medical Informatics Association*, page M2410v1, 2008.

38. D. Wigdor and D. Wixon. *Brave NUI world: designing natural user interfaces for touch and gesture*. Morgan Kaufmann, 2011.

39. J. Yang, K. Yu, Y. Gong, and T. Huang. Linear spatial pyramid matching using sparse coding for image classification. In *Computer Vision and Pattern Recognition (CVPRi)*, pages 1794–1801. IEEE, 2009.

40. R. Ziola, S. Grampurohit, N. Landes, J. Fogarty, and B. Harrison. Examining interaction with general-purpose object recognition in LEGO OASIS. In *Proc. of the IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC)*, 2011.