# Remote Object Fetching with Descriptive Question Asking

**Nathaniel Brennan**

A thesis presented for the degree of:

Bachelor of Science in Mathematics-Computer Science

Thesis Advisor:

Stefanie Tellex - Joukowsky Family Assistant Professor of Computer Science

Thesis Reader:

George Konidaris - Assistant Professor of Computer Science

Brown University

Providence, RI

May 1, 2019

## 0 Abstract

Human-robot communication is flawed because robots lack perfect knowledge of the world and human language is inherently imprecise. Therefore, we must develop novel frameworks for robots to overcome ambiguity in dialogue with humans. Given previous work by Whitney et al. [2017] towards asking simple questions to clarify intents in an object-fetching scenario using a POMDP, we describe a further extension to allow for clarifying dialogue with object attributes. Specifically, we define a model using predicates over attributes and their values as potential questions. This allows the model to clarify uncertainty efficiently. We show that this model is effective in a remote operation object-fetching scenario.

## 1 Introduction

In just the past ten years, applications of artificial intelligence and robotics have spread to a ever-growing diverse set of industries: robots are home cleaners, medical assistants, drivers, and more. However, in order to take advantage of the unique strengths of both people in their creativity and innovation and robots with their precision and ability to process massive amounts of data, robots must be able to successfully collaborate with human peers.

While voice assistants such as Google Home, Siri, and Alexa have found their way into the regular life of millions, such agents typically follow a one-way communication dialogue, with strictly the human operator prompting the assistant to take action. Furthermore, these dialogue systems often follow a one-directional tree of decisions, without room for backtracking or error correction. In order to allow robots to tackle more a complex set of tasks and ultimately act in seamless collaboration with humans, we look to provide a framework for the robot agent to prompt the operator as well, in order to efficiently clarify the desired task in the face of uncertainty. Object-picking tasks are found in almost all industries, from washing dishes, to picking olives or assembling cars. Using a natural language input space and constructing the robot's action space to optimize for object-picking, we are able to provide a novel dialogue framework focusing on optimizing the clarification capabilities of a robotic agent.

The POMDP model establishes a natural framework for handling uncertainty: it enables both question-asking when uncertain and acting when the task is clear. Following previous work by Whitney et al. [2017], we examine FETCH-POMDP as a starting point for using natural language to ask for objects and using clarifying questions as social feedback. One of the limitations of FETCH-POMDP, however, is its restriction to a single type of question for the robot. This type of question, called `point`, allows the robot to point to an object to ask the operator if this is the object desired. To give the robot more informative questions, we want to allow for clarifying questions about an ambiguously referenced object with regards to its specific attributes. We are motivated by our personal experience with using language to eliminate uncertainty for object picking tasks. When there are multiple spoons in a specific scene, when referring to a specific spoon, people are likely to communicate a specific spoon using its characteristics, such as ”the blue spoon”. With this in mind, our framework aims to include attribute-specifying questions to the robots' arsenal.

The object-picking scenario in Whitney et al. [2017] has the objects visible to both the user and the robot. In scenarios such as this, the user will typically describe their desired object with the necessary specificity to distinguish it from other objects. In the scenario above, the user will often say ”the blue spoon” without being prompted to say its color. Therefore, the robot's ability to ask questions about the attributes of the objects is not very useful in this scenario. Furthermore, gesture (such as the user pointing) is very informative and can eliminate the need for attribute questions in many interactions.

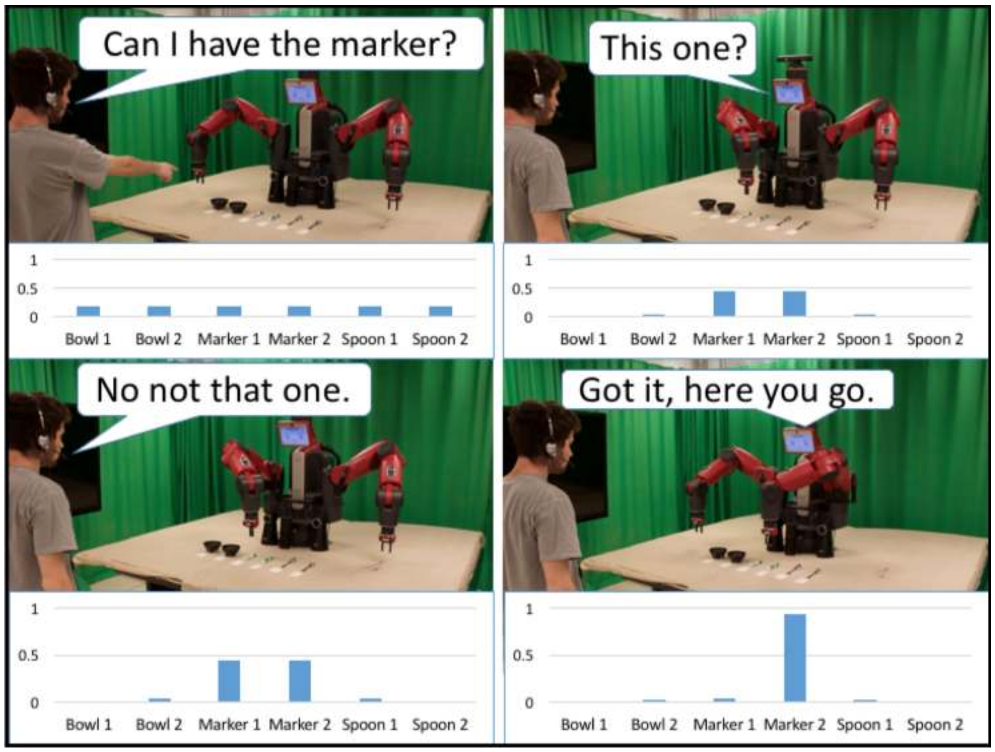
However, in an object-picking scenario where the user and robot aren't in the same room, the user can't see the set of objects that the robot is choosing between. Therefore, they might not be as specific as they need to be to distinguish their desired object from the other objects that the robot is comparing it to. If there are two spoons of different colors, the user might just say "Get me the spoon." In this case, it would be ideal for the robot to ask what color spoon they want, as this would lead to the fastest retrieval of the object. Therefore, we chose to apply our model to a remote object-fetching scenario rather than an in-person one.

We evaluated the model using a webapp interface and a scenario using six bowls with overlapping attributes. We had 15 subjects use the system for three interactions each, and we show that the system is effective for remote object-picking.

## 2 Related Works

Whitney et al. [2017] provides the most comprehensive and similar social feedback dialogue model for object picking. Based on this alone, question asking alongside the POMDP formulation for dialogue system has huge potential towards navigating the ambiguity of language in human-robot interactions. This work serves as the starting point for expanding the POMDP formulation towards navigating ambiguity with a more powerful set of questions.

Whitney et al. [2017] presents a POMDP model that defines the state as $(i_d, i_r)$, where $i_d$ is the desired object and $i_r$ is the last object asked about. The action space consists of three types of actions: `wait`, `point`, and `pick`, where $\texttt{point}_i$ asks if the user desires object $i$ and $\texttt{pick}_i$ retrieves an object $i$. This model observes both speech and gesture from the user. The FETCH-POMDP model defined rewards such that `wait` is the least costly, `point` is more costly, `pick`ing the wrong object is extremely costly, and `pick`ing the correct object is extremely rewarding.



They evaluated their model in two different 6-object scenarios: one with the objects spread far apart (the unambiguous configuration), and one with them close together (the ambiguous configuration). In both of these scenarios, the model was evaluated in terms of speed and accuracy against two baseline models: one that always asks a question (always-ask), and one that never asks a question (never-ask). In the ambiguous configuration, the model was not slower than the never-ask model, but significantly faster than the always-ask model. It was not more accurate than always-ask, but significantly more accurate than never-ask. In the unambiguous configuration, their model was faster than always-ask, and not faster than never-ask. There was no significant difference in accuracy in this scenario.

Deits et al. [2013] provides a base reference for creating an entropy-based heuristic for measuring uncertainty and determining the appropriate action a robotic agent should take. This approach could be used in the future to create an entropy heuristic-based solver for choosing the question that would reduce the uncertainty the most. This could potentially speed up computation time and improve accuracy compared to a POMDP solution.

Abbasnejad et al. [2018] also provides an interesting path forward by defining a Bayesian Deep Learning method for quantifying the uncertainty in the internal representation of a Reinforcement Learning model. They created an information-seeking decoder for a seq2seq question generation model. They evaluate their model on goal-oriented visual dialogue problems, which were a mix of cooperative and adversarial. They found their model outperformed baselines in both kinds of problems.

## 3 Technical Approach

### 3.1 POMDP Definition

Bellman [1957] introduces the Markov Decision Process (MDP), which is a model for decision-making at discrete time steps. At each time step, the agent observes the current state of the environment with perfect certainty, and takes an action, which affects the state at the next time step. An MDP is defined by $\langle S, A, T, R \rangle$ where $S$ is the set of environment states, $A$ is the set of possible actions the agent can take, $T(s, a, s')$ is the probability of transitioning to state $s'$ conditioned on the agent taking action $a$ in state $s$, and $R(s, a, s')$ is the reward for transitioning from $s$ to $s'$ after taking action $a$. The agent seeks to maximize the expected future reward.

A Partially Observable Markov Decision Process (POMDP) (described in Kaelbling et al. [1998]) is a generalization of an MDP which doesn't assume the state can be directly observed. Instead, at every time step the agent observes an element $o \in \Omega$, where $\Omega$ is the observation set. We assume that the observation at each timestep is taken from an observation function $O(o, s)$, which is the probability of observing $o$ conditioned on state $s$. We define a POMDP by $\langle S, A, T, R, \Omega, O \rangle$.

We also define $I$, our set of objects, $K$, the set of attributes that are defined for each object, and $V_k$, the values that an attribute $k \in K$ can have. For example, if we have a red and a blue marker, we would define $I = \{\texttt{R}, \texttt{B}\}$, $K = \{\texttt{color}\}$, and $V_{\texttt{color}} = \{\texttt{red}, \texttt{blue}\}$. We would say that R has attributes $\{\texttt{color} : \texttt{red}\}$ and B has attributes $\{\texttt{color} : \texttt{blue}\}$.

## 3.2 Action Space

The action set $A$ contains four different types of actions. The first is `wait`, which means that the agent does nothing for a short period of time before selecting a new action.

The second type of action is $\texttt{point}_i$ - the robot asks if the user desires a specific object. This is parameterized by an object $i \in I$.

Here, we introduce a second type of question - an attribute question, $\texttt{attr}_{k,v}$. This is parameterized by an attribute $k \in K$ and a value for this attribute $v \in V_k$. The agent will ask about this specific attribute-value pair: "Does attribute $k$ have value $v$?" In a scenario with two markers, one red and one blue, the robot might ask "is it blue?", where 'it' refers to the user's desired object.

The last action is $\texttt{pick}_i$ - retrieve an object $i \in I$. This action ends the interaction.

The action space is $A = \{\texttt{wait}\} \cup \{\texttt{point}_i\}_{i \in I} \cup \{\texttt{attr}_{k,v}\}_{k \in K, v \in V_k} \cup \{\texttt{pick}_i\}_{i \in I}$.

## 3.3 State Space

Our states are $(D, Q_t, Q_v) \in S$, where $D$ is the user's desired object, $Q_t$ is the type of question last asked by the robot, and $Q_v$ is the question value.

$Q_t \in \{\texttt{none}, \texttt{point}, \texttt{attr}\}$. If no question has been asked yet, then $Q_t = \texttt{none}$. If a question has been asked, then $Q_t$ is the type of the last question asked.

$Q_v$ takes on different types of values depending on the values of $Q_t$. If $Q_t = \texttt{none}$, then we assign $Q_v = \texttt{none}$. If $Q_t = \texttt{point}$, then $Q_v$ is the object that was asked about. If $Q_t = \texttt{attr}$, then $Q_v$ is a predicate that represents the attribute-value pair that the robot asked about. For example, if the robot asked if the color of the desired object was blue, then $Q_v = \lambda x \texttt{ color-blue}(x)$.

### 3.4 Transition Function

$D$ never changes. If a question is asked, $Q_t$ and $Q_v$ will change deterministically to reflect that.

### 3.5 Reward Function

We originally assigned a cost to `wait`, `point`, and `attr` based on the time each action takes. These values were then fine tuned to produce a smoother interaction. We set a high reward for a correct `pick` and a high cost for an incorrect `pick`.

### 3.6 Observation Model

Our observations are $(l_b, l_r) \in \Omega$. The response utterance $l_r$ contains all affirmative/negative words from the most recent speech detected, such as "yes", "yeah", "nope", etc. The base utterance $l_b$ contains all words that appear in $V = \cup_{k \in K} V_k$, the set of all values for all attributes.

We define our observation function by $O(o, s)$, which is the probability of observing $o \in \Omega$ conditioned on state $s \in S$. We assume that the observation is conditionally independent of the last action taken given $s$.

$$O(o, s) = \mathbb{P}(l_b, l_r \mid D, Q_t, Q_v) \tag{1}$$

We assume that $l_b$ and $l_r$ are conditionally independent given the state.

$$O(o, s) = \mathbb{P}(l_b \mid D, Q_t, Q_v)\mathbb{P}(l_r \mid D, Q_t, Q_v) \tag{2}$$

### 3.6.1 Base Utterance Model

We assume that $l_b$ is conditionally independent from $Q_t$ and $Q_v$ given $D$.

$$\mathbb{P}(l_b \mid D, Q_t, Q_v) = \mathbb{P}(l_b \mid D) \tag{3}$$

We then define the distribution as follows. $p_l$ is a fixed parameter representing the probability of observing an utterance. Recall that $l_b$ contains only the words that are in our vocabulary, $V$.

$$\mathbb{P}(l_b \mid D) = \begin{cases} p_l \prod_{w \in l_b} \mathbb{P}(w \mid D) & l_b \text{ is not empty} \\ 1 - p_l & l_b \text{ is empty} \end{cases} \tag{4}$$

We then define the distribution of individual words. Here, $V_D$ is the desired object's vocabulary, or the values that $D$ takes on for each attribute. $V = \cup_{k \in K} V_k$ is the set of all values for all attributes, and $\alpha$ is a fixed smoothing parameter.

$$\mathbb{P}(w \mid D) = \frac{\mathbb{1}_{V_D}(w) + \alpha}{|V_D| + \alpha |V|} \tag{5}$$

### 3.6.2 Response Utterance Model

We use a simplified model for $l_r$: we say it has to be positive, negative, or neither. If we observe an affirmative word, then $l_r = \texttt{yes}$. If we observe a negative word, then $l_r = \texttt{no}$. If we observe neither or both, then $l_r = \texttt{none}$.

We defined our set of affirmative words as {"yes", "yeah", "sure", "yup", "yep"} and our set of negative words as {"no", "nope", "not", "nah", "other"}.

When $Q_t = \texttt{none}$, we define $\mathbb{P}(l_r \mid D, Q_t, Q_v)$ to be uniform over $\{\texttt{yes}, \texttt{no}, \texttt{none}\}$.

For the following cases, we define a parameter $\epsilon$ that represents the small probability that the person doesn't answer a question in the way we assume they would - they lie, or there was a miscommunication that caused them to answer incorrectly. Recall that $p_l$ is the probability of observing an utterance.

For the case when $Q_t = \texttt{point}$, we define two different distributions for $\mathbb{P}(l_r \mid D, Q_t, Q_v)$ for the two cases when $Q_v = D$ and $Q_v \neq D$:

$$P(l_r \mid D, Q_t, Q_v) = \begin{cases} \begin{cases} p_l(1-\epsilon) & Q_v = D \\ p_l\epsilon & Q_v \neq D \end{cases} & l_r = \texttt{yes} \\ \begin{cases} p_l\epsilon & Q_v = D \\ p_l(1-\epsilon) & Q_v \neq D \end{cases} & l_r = \texttt{no} \\ \qquad 1 - p_l & l_r = \texttt{none} \end{cases} \tag{6}$$

Now we consider the case where $Q_t = \texttt{attr}$ and $Q_v = \lambda x \; \texttt{attr-val}(x)$ is the predicate that represents the attribute and value the robot asked about. We define two different distributions for $\mathbb{P}(l_r \mid D, Q_t, Q_v)$ for the two cases when $\texttt{attr-val}(D)$ is true (the desired object has that value for that attribute) and when it's false:

$$P(l_r \mid D, Q_t, Q_v) = \begin{cases} \begin{cases} p_l(1-\epsilon) & \texttt{attr-val}(D) \\ p_l\epsilon & \neg\texttt{attr-val}(D) \end{cases} & l_r = \texttt{yes} \\ \begin{cases} p_l\epsilon & \texttt{attr-val}(D) \\ p_l(1-\epsilon) & \neg\texttt{attr-val}(D) \end{cases} & l_r = \texttt{no} \\ \qquad 1 - p_l & l_r = \texttt{none} \end{cases} \tag{7}$$

## 4 Belief Update Tests

Here, we verify that our model works as expected by examining how the POMDP belief state reacts in a variety of scenarios. There are several variations that need to be made in order to cover the full scope of interactions: vary number of attributes, vary the question type, vary the utterance (observation), etc.

For all tests, we assume the initial belief distribution is uniform. We use parameters $\epsilon =$

0.01, $p_l = 0.95$, and $\alpha = 0.2$.

## 4.1 Two objects and one attribute

We have two objects, one red and one blue. Color is the only attribute.

$I = \{\texttt{R}, \texttt{B}\} \quad K = \{\texttt{color}\} \quad V_{\texttt{color}} = \{\texttt{red}, \texttt{blue}\} \quad s_0 = (D, \texttt{none}, \texttt{none})$

$z_0 =$"blue" $\quad l_b =$["blue"] $\quad l_r = \texttt{none}$

Result: $\quad b(\texttt{B}) = 0.857 \quad b(\texttt{R}) = 0.143$

## 4.2 Two objects and two attributes

The two attributes are color and orientation.

$K = \{\texttt{color}, \texttt{orientation}\} \quad V_{\texttt{color}} = \{\texttt{red}, \texttt{blue}\} \quad V_{\texttt{orientation}} = \{\texttt{x-aligned}, \texttt{y-aligned}\}$

$s_0 = (D, \texttt{none}, \texttt{none})$

### 4.2.1 "blue" with no overlap

We have two objects, one blue x-aligned and one red y-aligned.

$I = \{\texttt{ML}_{BX}, \texttt{ML}_{RY}\} \quad z_0$ = "blue" $\quad l_b =$["blue"] $\quad l_r = \texttt{none}$

**Result** $\quad b(\texttt{ML}_{BX}) = 0.857 \quad b(\texttt{ML}_{RY}) = 0.143$

### 4.2.2 "blue" with overlapping characteristic

We have two objects, one blue x-aligned and one red x-aligned.

$I = \{\texttt{ML}_{BX}, \texttt{ML}_{RX}\} \quad z_0$ = "blue" $\quad l_b =$["blue"] $\quad l_r = \texttt{none}$

**Result** $b(\texttt{ML}_{BX}) = 0.857 \quad b(\texttt{ML}_{RX}) = 0.143$

#### 4.2.3 "blue x-aligned" with no overlap

We have two objects, one blue x-aligned and one red y-aligned.

$I = \{\texttt{ML}_{BX}, \texttt{ML}_{RY}\}$ $z_0$ = "blue x-aligned" $l_b$ =["blue", "x-aligned"] $l_r = \texttt{none}$

**Result** $b(\texttt{ML}_{BX}) = 0.973 \quad b(\texttt{ML}_{RY}) = 0.027$

#### 4.2.4 "blue x-aligned" with overlapping characteristic

We have two objects, one blue x-aligned and one red x-aligned.

$I = \{\texttt{ML}_{BX}, \texttt{ML}_{RX}\}$ $z_0$ = "blue x-aligned" $l_b$ =["blue", "x-aligned"] $l_r = \texttt{none}$

**Result** $b(\texttt{ML}_{BX}) = 0.857 \quad b(\texttt{ML}_{RX}) = 0.143$

## 4.3 Point question

We have two objects, one red and one blue. The robot has just pointed at $\texttt{ML}_B$.

$I = \{\texttt{ML}_R, \texttt{ML}_B\} \quad K = \{\texttt{color}\} \quad V_{\texttt{color}} = \{\texttt{red}, \texttt{blue}\} \quad s_0 = (D, \texttt{point}, \texttt{ML}_B)$

#### 4.3.1 "yes"

$z_0$ = "yes" $l_b$ = [] $l_r = \texttt{yes}$

**Result** $b(\texttt{ML}_B) = 0.99 \quad b(\texttt{ML}_R) = 0.01$

#### 4.3.2 "yes the blue one"

$z_0$ = "yes the blue one" $\quad l_b$ = ["blue"] $\quad l_r = \texttt{yes}$

**Result** $\quad b(\texttt{ML}_B) = 0.998 \quad b(\texttt{ML}_R) = 0.002$

#### 4.3.3 "no"

$z_0$ = "no" $\quad l_b$ = [] $\quad l_r = \texttt{no}$

**Result** $\quad b(\texttt{ML}_B) = 0.01 \quad b(\texttt{ML}_R) = 0.99$

#### 4.3.4 "no not the blue one"

$z_0$ = "no not the blue one" $\quad l_b$ = ["blue"] $\quad l_r = \texttt{no}$

**Result** $\quad b(\texttt{ML}_B) = 0.057 \quad b(\texttt{ML}_R) = 0.943$

#### 4.3.5 "no the red one"

$z_0$ = "no the red one" $\quad l_b$ = ["red"] $\quad l_r = \texttt{no}$

**Result** $\quad b(\texttt{ML}_B) = 0.002 \quad b(\texttt{ML}_R) = 0.998$

### 4.4 Attribute question

If we use only two objects, this case is the same as above. So we will only test it on the case where there are three objects, and two of them have the same color.

$I = \{\texttt{ML}_{R1}, \texttt{ML}_{R2}, \texttt{ML}_B\} \quad K = \{\texttt{color}\} \quad V_{\texttt{color}} = \{\texttt{red}, \texttt{blue}\} \quad s_0 = (D, \texttt{attr}, \texttt{color-blue}(\text{x}))$

#### 4.4.1 “yes”

$z_0 =$“yes” $l_b$ = [] $l_r = \texttt{yes}$

**Result** $b(\texttt{ML}_{R1}) = 0.010$ $b(\texttt{ML}_{R2}) = 0.010$ $b(\texttt{ML}_{B}) = 0.980$

#### 4.4.2 “yes the blue one”

$z_0 =$“yes the blue one” $l_b$ = [“blue”] $l_r = \texttt{yes}$

**Result** $b(\texttt{ML}_{R1}) = 0.002$ $b(\texttt{ML}_{R2}) = 0.002$ $b(\texttt{ML}_{B}) = 0.997$

#### 4.4.3 “no”

$z_0 =$“no”. $l_b$ = [] $l_r = \texttt{no}$

**Result** $b(\texttt{ML}_{R1}) = 0.497$ $b(\texttt{ML}_{R2}) = 0.497$ $b(\texttt{ML}_{B}) = 0.005$

#### 4.4.4 “no not the blue one”

$z_0 =$“no not the blue one”. $l_b$ = [“blue”] $l_r = \texttt{no}$

**Result** $b(\texttt{ML}_{R1}) = 0.485$ $b(\texttt{ML}_{R2}) = 0.485$ $b(\texttt{ML}_{B}) = 0.029$

#### 4.4.5 “no the red one”

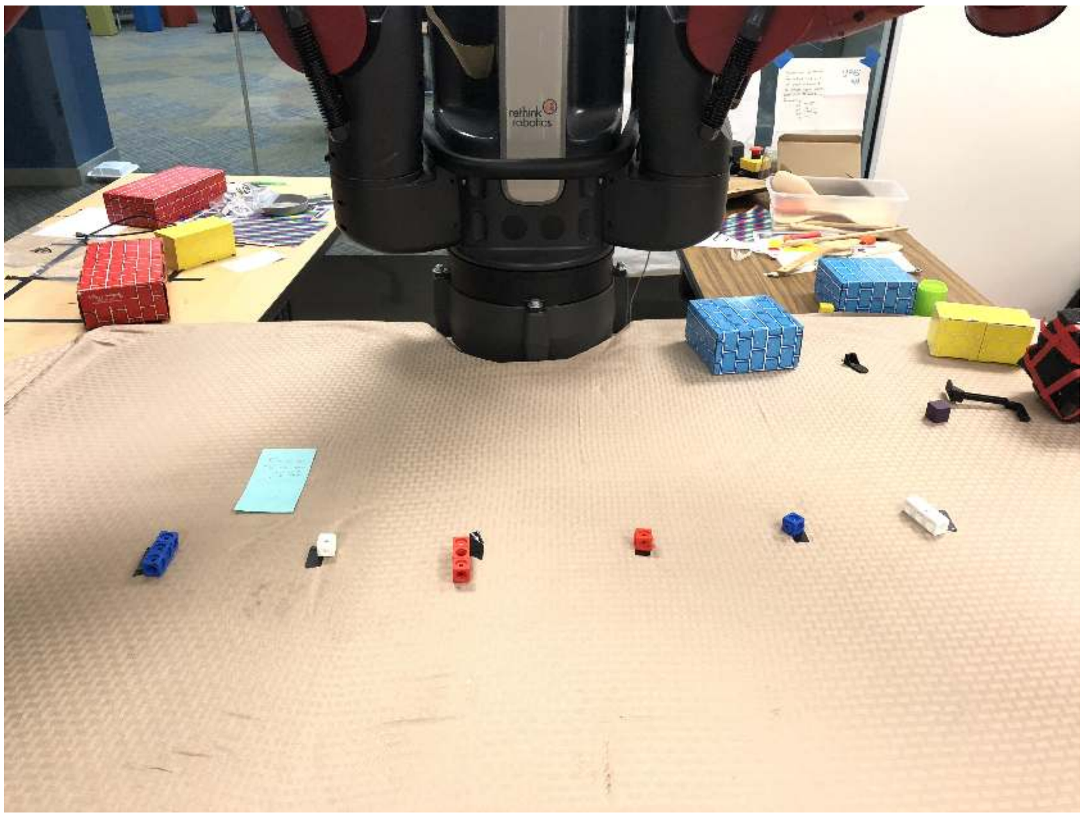
$z_0 =$“no the red one”. $l_b$ = [“red”] $l_r = \texttt{no}$

**Result** $b(\texttt{ML}_{R1}) = 0.500$ $b(\texttt{ML}_{R2}) = 0.500$ $b(\texttt{ML}_{B}) = 0.001$

The belief distribution reacts as expected in all scenarios.

## 5 Initial Evaluation

We evaluated the model defined above with a setup similar to that in Whitney et al. [2017]. We used the Baxter robot from Rethink Robotics as the robotic agent, and used a few random students as test subjects. We set up six objects in front of Baxter:



To validate our hypothesis that attribute questions would reduce the speed of these interactions, we compared the performance of our model in this scenario with that of a baseline model. The baseline model is equivalent to our model except it cannot ask attribute questions: it only has access to the `wait`, `point`, and `pick` actions.

To evaluate the performance of our model and compare it to the baseline model, we specifically looked at the ability for each model to clarify uncertainty from an initial confused state. This means that the robot begins the interaction with a clarifying question, rather than the user beginning the interaction with an instruction.

Given that robotic agents will almost inevitably encounter situations in which the user agent provided information is incomplete, not well defined, or otherwise not understood perfectly by the robotic agent, we sought to highlight the differences in each model's ability to provide meaningful social feedback. This way, we specifically highlight and examine social feedback capabilities.

With this in mind, we define the start of an interaction as the time when Baxter begins its first action and the end of the interaction as the time when Baxter finishes handing off the object to the user. Furthermore, we define a correct `pick` as a handoff of the user's desired object, and all other handoffs as an incorrect `pick`.

We tested both models with three subjects. With the first subject, our model took 63 seconds and the baseline model took 29. For the second subject, our model took 57s and the baseline model took 33s. With the third subject, our model took 62s and the baseline model took 91s. Furthermore, we note that in both models and across all three test subjects, the desired object was always correctly identified.

From these preliminary results, we can see that there is no guaranteed improvement from using attribute questions in this given interaction. We can clearly note though, given a user that answers correctly to all prompts, the upper bounds for questions needed in our model is an improvement over that of the baseline in this scenario. Under the given circumstances, our model would need to ask at most three questions to correctly identify a desired object, while the baseline would need to ask at most five.

However, this theoretical guarantee does not accurately describe the difference between the two models. In a scenario where all the objects are visible to the user, we observed that users will always be as specific as they need to be with their instructions in order to differentiate their desired objects from the others. The language they use to describe their object will only be an accurate description of that object and no others. If there are two bowls that are identical except one is slightly bigger than the other, the user will say "get me the bigger bowl" in order to specify which of the two bowls they want. This means that the attribute questions were not as useful as we originally thought in this scenario, because often they aren't needed at all.

## 6 Remote Operation

After these disappointing results, we looked for a scenario in which attribute questions would be useful, or necessary, to complete an object-picking interaction. We decided on a scenario in which the robot and the user aren't in the same room, and the user is operating it remotely.

Imagine you have a robot assistant, and you need it to retrieve an object in another room. You know what object you want, but you don't know what other objects may be in the other room. The robot will be comparing many different objects with different characteristics, but you have no knowledge of what characteristics distinguish your desired object from other objects in the room.

In this scenario, attribute questions have a clear advantage. When the user isn't specific enough for the robot to narrow down their search to a single object, the robot can use its knowledge of the objects to ask an intelligent question to gain the information that the user failed to provide initially.

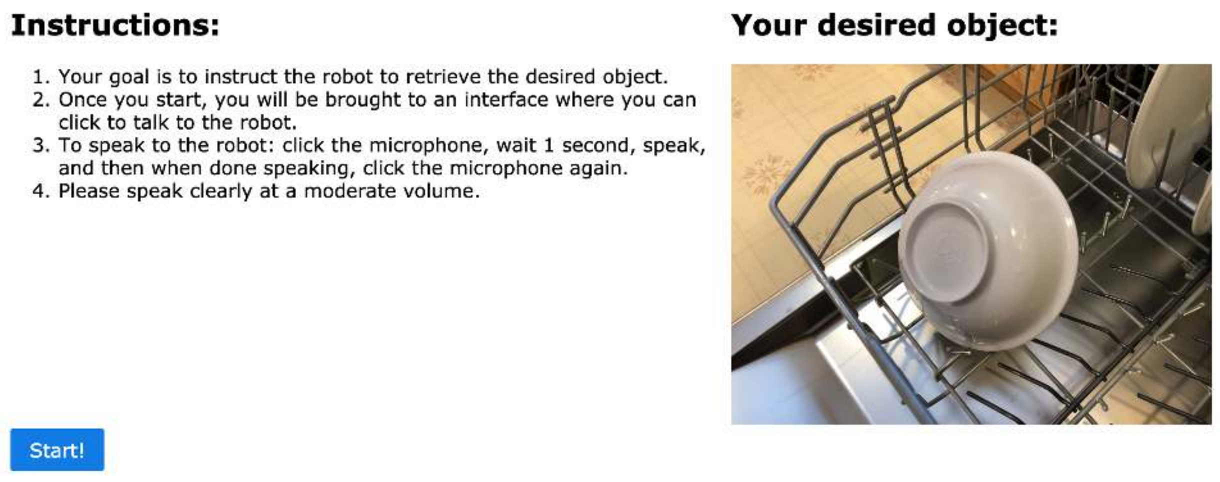
For this scenario, we decided to use six bowls as our objects. The bowls have overlapping characteristics, so the robot will be likely to face ambiguity in an average interaction.

For this scenario we defined $I = \{$`white`, `white2`, `white3`, `metal`, `metal2`, `clear`$\}$. The attributes are $K = \{$`color`, `material`, `size`, `shape`, `location`$\}$. Here are the full definitions of the objects. Note that `shape` was included only to test whether the agent would ever choose to ask about it (it did not).

| | **color** | **material** | **size** | **shape** | **location** |
|---|---|---|---|---|---|
| **white** | white | plastic | small | round | dishwasher |
| **white2** | white | plastic | small | round | counter |
| **white3** | white | plastic | big | round | cabinet |
| **metal** | metallic | metal | small | round | cabinet |
| **metal2** | metallic | metal | big | round | counter |
| **clear** | clear | plastic | big | round | counter |

We realized to test this scenario, we didn't have to use a physical robot. Instead, we could simulate it using a web interface. We developed one that allowed the user to speak to the

”robot” and had the robot speak back.



**Instructions:**

1. Your goal is to instruct the robot to retrieve the desired object.
2. Once you start, you will be brought to an interface where you can click to talk to the robot.
3. To speak to the robot: click the microphone, wait 1 second, speak, and then when done speaking, click the microphone again.
4. Please speak clearly at a moderate volume.

**Your desired object:**

In each interaction, the program chooses a bowl uniformly at random, and asks the user to instruct the robot to retrieve it. We removed the `point` action here because we had no way of translating it to this scenario.

We conducted user trials with this web interface. We had 15 participants, and we had each of them complete three interactions. Out of the 45 total interactions, the system retrieved the correct object in 36 of them, for a success rate of **80%**. The average interaction time was **35.8 seconds**.

We noted that many of the failed interactions were due to poor timing, such as when the user answers the robot’s question just as the robot is starting to ask a second one. This causes the robot to interpret the user’s answer to the first question as their answer to the second question, which leads to the robot picking the wrong object.

## 7 Conclusion

In this paper we present a new model for reducing confusion by asking clarifying questions during object fetching. Our model is based on the model described in Whitney et al.

[2017], but we focus on descriptive attribute questions rather than the robot pointing at objects or the user gesturing. We find that the POMDP is still solvable with this expanded state-action space.

There are many ways to build on this model. We could give the robot the ability to ask questions of the form: "What the value is of attribute $k$?" There are many ways the language model could be improved, since the one we use here is simple. The system could be augmented to include intelligent question generation like the system described in Abbasnejad et al. [2018]. Questions could be chosen with an entropy-based heuristic like the one described in Deits et al. [2013].

This model can be applied to a variety of other decision making tasks. Any home robot will need to perform complex tasks in order to be useful, and even with the current state of the art in natural language processing and world modeling, the robot can easily become confused. When a human user can't clarify the ambiguity because they are unaware of what the robot is confused about, robots can still use their own knowledge to produce questions to clarify meaning. This will lead to more successful human-robot interactions.

## 8 Acknowledgements

I would like to thank Christopher Chen for his hard work on this project.

I would like to thank my advisor Stefanie Tellex, as well as Daniel Ullman, Matt Corsaro and Eric Rosen for their guidance and advice.

I would also like to thank George Konidaris for serving as my reader.

## References

David Whitney, Eric Rosen, James MacGlashan, Lawson L.S. Wong, and Stefanie Tellex. Reducing errors in object-fetching interactions through social feedback. *International Conference on Robotics and Automation*, 2017.

Robin Deits, Stefanie Tellex, Pratiksha Thaker, Dimitar Simeonov, Thomas Kollar, and Nicholas Roy. Clarifying commands with information-theoretic human-robot dialog. *Journal of Human-Robot Interaction*, 2013.

Ehsan Abbasnejad, Qi Wu, Javen Shi, and Anton van den Hengel. What's to know? uncertainty as a guide to asking goal-oriented questions. 2018.

Richard Bellman. A markovian decision process. *Journal of Mathematics and Mechanics*, 6(5):679–684, 1957. ISSN 00959057, 19435274. URL http://www.jstor.org/stable/24900506.

Leslie Pack Kaelbling, Michael L. Littman, and Anthony R. Cassandra. Planning and acting in partially observable stochastic domains. *Artif. Intell.*, 101(1-2):99–134, May 1998. ISSN 0004-3702. doi: 10.1016/S0004-3702(98)00023-X. URL http://dx.doi.org/10.1016/S0004-3702(98)00023-X.