Linguistic Theories in Efficient Multimodal Reference Resolution: An Empirical Investigation

Joyce Y. Chai Zahar Prasov Joseph Blaim Rong Jin
Department of Computer Science and Engineering
Michigan State University
East Lansing, MI 48824
{jchai, prasovza, blaimjos, rongjin} @cse.msu.edu

ABSTRACT

Multimodal conversational interfaces provide a natural means for users to communicate with computer systems through multiple modalities such as speech, gesture, and gaze. To build effective multimodal interfaces, understanding user multimodal inputs is important. Previous linguistic and cognitive studies indicate that user language behavior does not occur randomly, but rather follows certain linguistic and cognitive principles. Therefore, this paper investigates the use of linguistic theories in multimodal interpretation. In particular, we present a greedy algorithm that incorporates *Conversation Implicature* and *Givenness Hierarchy* for efficient multimodal reference resolution. Empirical studies indicate that this algorithm significantly reduces the complexity in multimodal reference resolution compared to a previous graph-matching approach. One major advantage of this greedy algorithm is that the prior linguistic and cognitive knowledge can be used to guide the search and significantly prune the search space. Because of its simplicity and generality, this approach has the potential to improve the robustness of interpretation and provide a more practical solution to multimodal input interpretation.

Categories & Subject Descriptors: H.5.2 (User Interfaces): Theory and method, Natural language

General Terms: Algorithms, Design, Experimentation

Keywords: Multimodal input interpretation, reference resolution

1. INTRODUCTION

Multimodal user interfaces enable users to interact with computers naturally and efficiently through multiple modalities such as speech, gesture, and gaze [15, 17]. One important aspect of multimodal conversational interfaces is the capability to identify entities that users refer to in their multimodal inputs, in other words, multimodal reference resolution. In a multimodal conversation, the way users communicate with a system depends on the available interaction channels and the situated context (e.g., conversation focus, visual feedback). These dependencies form a rich set of constraints from various perspectives such as temporal alignments between different modalities, coherence of conversation, and domain semantics.

To obtain the most probable interpretation based on these constraints, an optimization approach using probabilistic graph matching was developed [1, 3]. This approach has its theoretical merit since it aims for a global optimization when matching referring expressions to potential referents based on a set of constraints. However, as most optimization approaches, the graph-matching algorithm has a non-polynomial (NP) nature. It could become intractable once the number of referring expressions and the number of potential referents (e.g., objects on the screen) are increased. Thus a more practical solution is desired.

Previous linguistic and cognitive studies indicate that user referring behavior does not occur randomly, but rather follows certain linguistic and cognitive principles. Our hypothesis is that prior knowledge from these studies can be used to guide the matching process and reduce the complexity in constraint satisfaction. With this in mind, the focus of this paper is to empirically investigate the use of linguistic theories in efficient multimodal reference resolution. Specifically, we investigate two linguistic theories: *Conversation Implicature* and *Givenness Hierarchy*. We present a greedy algorithm that utilizes these two theories. Given *m* referring expressions and *n* potential referents from various sources (e.g., gesture, conversation context, and visual display), this algorithm can find a solution in *O(mn)*. Empirical studies indicate that this algorithm achieves comparable performance to the graph-matching approach. One major advantage of this greedy algorithm is that the prior linguistic and cognitive knowledge can be used to guide the search and significantly prune the search space. Because of its simplicity and generality, this approach has the potential to improve the robustness of interpretation and provide a more practical solution to multimodal reference resolution.

In the following sections, we first introduce two linguistic theories and then describe how they can be used to design efficient algorithms for multimodal reference resolution.

IUI'05, January 9–12, 2005, San Diego, California, USA

Status:	*In focus* >	*activated* >	*familiar* >	*uniquely identifiable* >	*referential* >	*identifiable*
Expression form:	(it)	(that, this, this N)	(that N)	(the N)	(indefinite this N)	(a N)

Figure 1: Givenness Hierarchy

2. RELATED WORK

Considerable work has been done on studying multimodal referring behavior [5, 16] and mechanisms to resolve multimodal referring expressions [3, 4, 6, 10, 12, 13, 14, 19, 20]. In multimodal reference resolution, two issues are important: (1) combining information from various sources to form an overall interpretation given a set of constraints, and (2) obtaining the best interpretation among all the possible alternatives given a set of constraints.

Much of the earlier work has been focused on the first issue, for example, using the centering framework [20] and contextual factors [10]. One major mechanism is multimodal fusion, for example, through unification-based approaches [12] or finite state approaches [13]. The unification-based approach identifies referents to referring expressions by unifying feature structures generated from speech utterances and from gestures based on a multimodal grammar [11, 12]. The grammar rules are predefined based on empirical studies of multimodal interaction [16]. For example, one rule indicates that speech and gesture can be combined only when the speech either overlaps with gesture or follows the gesture within a certain time frame. If a specific user referring behavior does not exactly match any existing integration rules (e.g., temporal relations), the unification fails and therefore references are not resolved. In the finite state approach [13], a multimodal context-free grammar is defined to transform the syntax of the multimodal inputs into their semantic meanings. The domain specific semantics are directly encoded in the grammar. Based on these grammars, multi-tape finite state automata can be constructed. These automata are used for identifying semantics of combined inputs.

To address the second issue, an optimization approach was developed that uses a graph-matching algorithm for multimodal reference resolution [1, 3]. In this approach, information gathered from multiple input modalities and the conversation context is represented as attributed relational graphs. Specifically, one graph represents the semantic and temporal information for referring expressions and their semantic and temporal relations; and the other graph represents all potential referents and their semantic and temporal relations. Given the semantic and temporal constraints modeled in these graphs, the multimodal reference resolution problem becomes a probabilistic graph-matching problem that identifies the most compatible match between two graphs. Theoretically, this approach provides a solution that maximizes the overall satisfaction of semantic, temporal, and contextual constraints. However, like many other optimization approaches, this algorithm is non-polynomial (NP). It relies on an expensive matching process, which attempts every possible assignment, in order to converge on an optimal interpretation based on those constraints. The question arises whether any information can be used to guide this matching process and reduce the complexity. Therefore, we investigate linguistic theories for a potential solution.

3. LINGUISTIC THEORIES

To investigate the use of linguistic and cognitive knowledge in efficient multimodal reference resolution, we specifically focus on two theories: *Conversation Implicature* and *Givenness Hierarchy*.

3.1 Conversation Implicature

Grice's Conversation Implicature Theory indicates that the interpretation and inference of an utterance during communication is guided by a set of four maxims [7]. Among these four maxims, the *Maxim of Quantity* and the *Maxim of Manner* are particularly useful for our purpose.

The Maxim of Quantity has two components: (1) make your contribution as informative as is required (for the current purposes of the exchange), and (2) do not make your contribution more informative than is required. In the context of multimodal conversation, this maxim indicates that users generally will not make any unnecessary gestures or speech. This is especially true for pen-based gestures since they usually take a special effort from a user. Therefore, when a pen-based gesture is intentionally delivered by a user, the information conveyed is often a crucial component used in interpretation.

Grice's Maxim of Manner has four components: (1) avoid obscurity of expression, (2) avoid ambiguity, (3) be brief, and (4) be orderly. This maxim indicates that users will not intentionally make ambiguous references. They will use expressions (either speech or gesture) they believe can uniquely describe the object of interest so that listeners (in this case a computer system) can understand. The expressions they choose depend on the information in their mental models about the current state of the conversation. However, the information in a user's mental model might be different from the information the system possesses. When such an information gap happens, different ambiguities could occur from the system point of view. In fact, most ambiguities are not intentionally caused by the human speakers, but rather by the system's incapability of choosing among alternatives given incomplete knowledge representation, limited capability of contextual inference, and other factors (e.g., interface design issues). Therefore, the system should not anticipate deliberate ambiguities from users (e.g., a user only utters "a house" to refer to a particular house on the screen), but rather should focus on dealing with the types of ambiguities caused by the system's limitations (e.g., gesture ambiguity due to the interface design or speech ambiguity due to incorrect recognition).

3.2 Givenness Hierarchy

The Givenness Hierarchy proposed by Gundel et al. explains

how different determiners and pronominal forms signal different information about memory and attention state (i.e., cognitive status) [9]. As in Figure 1, there are six cognitive statuses in the hierarchy. For example, Focus indicates the highest attentional state that is likely to continue to be the topic. Activated indicates entities in short term memory. Each of these statuses is associated with some forms of referring expressions. In this hierarchy, each cognitive status implies the statuses to its right. For example, "*in focus*" implies "*activated*", "*familiar*", etc. The use of a particular expression form not only signals that the associated cognitive status is met, but also signals that all lower statuses have been met. In other words, a given form that is used to describe a lower status can also be used to refer to a higher status, but not vice versa. Cognitive statuses are necessary conditions for appropriate use of different forms of referring expressions. Gundel et al. found that different referring expressions almost exclusively correlate with the six statuses in this hierarchy.

A previous investigation on the Givenness Hierarchy in multimodal interaction was reported in [14]. Based on data collected from Wizard of Oz experiments, this investigation suggests that users tend to tailor their expressions to what they perceive to be the system's beliefs concerning the cognitive status of referents from their prominence (e.g., highlight) on the display. The tailored referring expressions can then be resolved with a high accuracy based on the following decision list:

1. If an object is gestured to, choose that object.
2. Otherwise, if the currently selected object meets all semantic type constraints imposed by the referring expression, choose that object.
3. Otherwise, if there is a visible object that is semantically compatible, then choose that object.
4. Otherwise, a full NP (such as a proper name) is used to uniquely identify the referent.

From our studies [2], We found this decision list has the following limitations:

- Depending on the interface design, ambiguities (from a system's perspective) could occur. For example, given an interface where one object (e.g., house) can be sometimes created on top of another object (e.g., town), a pointing gesture could result in multiple potential object references. Furthermore, given an interface with crowded objects, a finger point could also result in multiple potential references. The decision list is not able to handle these "ambiguous cases".
- User inputs are not always simple (consisting of no more than one referring expression and one gesture as indicated in the decision list). In fact, in our study [2], we found that 23% user inputs are complex inputs (consisting of multiple referring expressions and/or multiple gestures). The referents to these referring expressions could come from different sources, such as gesture inputs and conversation context. The temporal alignment between speech and gesture is also important in determining the correct referent for a given expression. The decision list is not able to handle these types of complex inputs.

Nevertheless, the findings from [14] have inspired our work described in this paper. In particular, we would like to extend the previous work and investigate whether Conversation Implicature and Givenness Hierarchy can be used to resolve a variety of references from simple to complex, and from precise to ambiguous. Furthermore, the decision list in [14] is proposed based on data analysis and has not been implemented or evaluated in a real-time system. Therefore, one of our goals is to design and implement an efficient algorithm by incorporating these linguistic theories and empirically compare its performance with the optimization approach described in [1].

4. A GREEDY ALGORITHM

A greedy algorithm always makes the choice that looks best at the moment of processing. That is, it makes a locally optimal choice in the hope that this choice will lead to a globally optimal solution. Simple and efficient greedy algorithms can be used to approximate many optimization problems. Here we explore the use of Conversation Implicature and Givenness Hierarchy in designing an efficient greedy algorithm. In particular, we utilize the concepts from the two linguistic theories in the following way:

(1) Corresponding to the Givenness Hierarchy, the following hierarchy holds for potential referents: *Focus* > *Visible*. This hierarchy indicates that objects in the focus have higher status in terms of attention states than objects in the visual display. Here *Focus* corresponds to the cognitive statuses "*in focus*" and "*activated*" in the Givenness Hierarchy, and *Visible* corresponds to the statuses "*familiar*" and *"uniquely identifiable"*.

(2) Based on the Conversation Implicature, since a pen-based gesture takes a special effort to deliver, it must convey certain information. In fact, objects indicated by a gesture should have the highest attentional state since they are intentionally singled out by a user.

Therefore, by combining (1) and (2), we derive a *modified hierarchy* ***Gesture > Focus > Visible > Others***. Here *Others* corresponds to indefinite cases. This hierarchy coincides with the processing order of the decision list in [14]. This modified hierarchy will guide the greedy algorithm in its search for solutions. Next, we describe in detail the algorithm and related representations and functions.

4.1 Representation

At each turn [1] (e.g., after receiving a user input) of the conversation, we use three vectors to represent the first three statuses in our modified hierarchy: objects selected by a gesture, objects in the focus, and objects visible on the display as follows:

- Gesture vector ($\vec{g}$) captures objects selected by a series of gestures. Each element g_i is an object selected by a gesture. For elements g_i and g_j where $i < j$, the gesture that selects objects g_i should either precede (temporally) or the same as the gesture that selects g_j.

[1] Currently, user inactivity (i.e., 2 seconds with no input from either speech or gesture) is used as the boundary to decide an interaction turn.

- Focus vector ($\vec{f}$) captures objects that are in the focus but are not selected by any gesture. Each element represents an object that is the focus of attention from the previous turn of the conversation. There is no temporal precedence relation between these elements. We consider all the corresponding objects are simultaneously accessible to the current turn of the conversation.
- Display vector ($\vec{d}$) captures objects that are visible on the display but are neither selected by any gesture (e.g. $\vec{g}$) nor in the focus ($\vec{f}$). There is also no temporal precedence relation between these elements. All elements are simultaneously accessible.

Based on these representations, every object on the display belongs to one and only one vector. Each object consists of the following information:

- Semantic type of the object. For example, whether the object is a House or a Town.
- The attributes of the object. This is a domain dependent feature. A set of attributes is associated with each semantic type. For example, a house object has Price, Size, Year of Built, etc. as its attributes. Furthermore, each object has visual properties that reflect the appearance of the object on the display such as Color of an object icon.
- The identifier of the object. Each object has a unique name.
- The selection probability. It refers to the probability that a given object is selected. Depending on the interface design, a gesture could result in a list of potential referents. We use this selection probability to indicate the likelihood of an object selected by a gesture. The calculation of the selection probability is described in [3]. For objects from the focus vector and the display vector, the selection probabilities are set to 1/N where N is the total number of objects in the respective vector.
- Temporal information. The relative temporal ordering information for the corresponding gesture. Instead of applying time stamps as in [3], here we only use the index of gestures according to the order of their occurrences. If an object is selected by the first gesture, then its temporal information would be "1".

In addition to vectors that capture potential referents, at each turn, a vector that represents referring expressions from a speech utterance ($\vec{r}$) is also maintained. Each element (i.e., a referring expression) has the following information:

- The identifier of the potential referent indicated by the referring expression. For example, the identifier of the potential referent to the expression "house number eight" is a house object with an identifier Eight.
- The semantic type of the potential referents indicated by the expression. For example, the semantic type of the referring expression "this house" is House.
- The number of potential referents as indicated by the referring expression or the utterance context. For example, a singular noun phrase refers to one object. A phrase like "three houses" provides the exact number of referents (i.e., 3).
- Type dependent features. Any features, such as Color and Price associated with potential referents, are extracted from the referring expression.
- The temporal ordering information indicating the order of referring expressions as they are uttered. Again, instead of the specific time stamp as in [3], here we only use the temporal ordering information. If an utterance consists of N consecutive referring expressions, then the temporal ordering information for each of them would be 1, 2, and up to N.
- The syntactic categories of the referring expressions. Currently, for each referring expression, we assign it to one of six syntactic categories (e.g., demonstrative and pronoun). Details are explained later.

These four vectors are updated after each user turn in the conversation based on the current user input and the system state (e.g., what is shown on the screen and what was identified as focus from the previous turn of the conversation).

4.2 Algorithm

The pseudo code for the algorithm is shown in Figure 2. For each multimodal input at a particular turn in the conversation, this algorithm takes the inputs of a vector ($\vec{r}$) of referring expressions with size *k*, a gesture vector ($\vec{g}$) of size *m*, a focus vector of ($\vec{f}$) of size *n*, and a display vector ($\vec{d}$) of size *l*. It first creates three matrices to capture the scores of matching each referring expression from $\vec{r}$ to each object in the other three vectors. Calculation of the matching score is described later.

Based on the matching scores in the three matrices, the algorithm applies a greedy search that is guided by our modified hierarchy as described earlier. Since *Gesture* has the highest status, the algorithm first searches the Gesture Matrix (G) that keeps track of matching scores between all referring expression and all objects from gestures. It identifies the highest (or multiple highest) matching scores and assigns all possible objects from gestures to the expressions (GreedySortingGesture).

If more referring expressions are left to be resolved after gestures are processed, the algorithm looks at objects from the Focus Matrix (F) since *Focus* is the next highest cognitive status (GreedySortingFocus). If there are still more expressions to be resolved, then the algorithm looks at objects from the Display Matrix (D) (GreedySortingDisplay). Currently, our algorithm focuses on these three statuses. Certainly, if there are still more expressions to be resolved after all these steps, the algorithm can consult with proper name resolution. Once all the referring expressions are resolved, the system will output the results. For the next multimodal input, the system will generate four new vectors and then apply the greedy algorithm again.

Note that in GreedySortingGesture, we use index-max to keep track of the column index that corresponds to the largest matching value. As the algorithm incrementally processes each row in the matrix, this index-max should incrementally increases. This is

because the referring expressions and the gesture should be aligned according to their order of occurrences. Since objects in the Focus Matrix and the Display Matrix do not have temporal precedence relations, GreedySortingFocus and GreedySortingDisplay do not use this constraint.

```
GreedyMultimodalReferenceResolution ( g⃗ , f⃗ , d⃗ , r⃗ )
   InitializeMatchMatrix( g⃗ , f⃗ , d⃗ , r⃗ )
   If G is not empty  // there are one or more gestures
   Then {GreedySortingGesture
         If (all referring expressions in r⃗ are resolved)
         Then exit}
   If F is not empty
   Then  {GreedySortingFocus
          If (all referring expressions in r⃗ are resolved)
          Then exit}
   GreedySortingDisplay
}

InitializeMatchMatrix ( g⃗ , f⃗ , d⃗ , r⃗ ){
  for (i = 1..m; j = 1..k)   G[i][j] = Match(g_i, r_j)
  for (i = 1..n; j = 1..k)   F[i][j] = Match(f_i, r_j)
  for (i = 1..l; j = 1..k)   D[i][j] = Match(d_i, r_j)
}

GreedySortingGesture {
 index_max = 1; //index to the column
 for (i = 1..m) {
  find j ≥ index_max, where G[i][j] is the largest
            among the elements in row i.
     add a mark "*" to the cell G[i][j];
     index_max = j;
 } //complete finding the best match from a view of each object
 AssignReferentsFromMatrix (G);
}

GreedySortingFocus{
   for (j = 1..k)
     if (r_j is resolved)
     then Cross out column j in F  //only keep ones not resolved
   for ( i = 1..n){
     find j where F[i][j] is the largest among the elements in row i.
     mark "*" to the cell F[i][j]; }
   AssignReferentsFromMatrix (F);
}

GreedySortingDisplay{
   for (j = 1..k)
     if (r_j is resolved)
     then Cross out column j in D;
   for ( i = 1..l){
     find j where D[i][j] is the largest among the elements in row i.
     mark "*" to D[i][j]; }
   AssignReferentsFromMatrix (D);
}

AssignReferentsFromMatrix (Matrix X){
for (i = 1..k) // i.e., for each expression r_i  in column i
  if (r_i indicates a specific number N and more than N elements in
      ith column of X with "*")
  then assign N largest elements  with "*"  to r_i as referents.
  else assign all elements with "*" to r_i as referents
}
```

Figure 2: Pseudo code of the greedy algorithm

This greedy algorithm also applies the dynamic programming principle. Each object (no matter whether it is introduced to the discourse from gesture, previous conversation, or simply the graphic display) first finds its best match to the referring expressions. Such a match is recorded through "*" for each object. Then at the global level, each referring expression will find its best matches based on the order of our modified hierarchy. The reason we call this algorithm "greedy algorithm" is that it always finds the best assignment for a referring expression given a cognitive status in the hierarchy. In other words, this algorithm always makes the best choice for each referring expression one at a time according to the order of their occurrence in the utterance. One can imagine that, a mistaken assignment made to an expression can affect the assignment of the following expressions. Therefore, the greedy algorithm may not lead to a globally optimal solution. Nevertheless, the general user behavior following the guiding principles makes this greedy algorithm useful.

One major advantage of this greedy algorithm is that the use of the modified hierarchy significantly prunes the search space compared to the graph-matching approach. Given *m* referring expressions and *n* potential referents from various sources (e.g., gesture, conversation context, and visual display), this algorithm can find a solution in *O(mn)*. Furthermore, this algorithm goes beyond simple and precise inputs as illustrated by the decision list in [14]. The scoring mechanism (described later) and the greedy sorting process accommodate both complex and ambiguous user inputs.

4.3 Matching Function

An important component of the algorithm is the matching score between an object (*o*) and a referring expression (*e*). We use the following formula to calculate such a score:

$$Match(o,e) = [\sum_{S \in \{G,F,D\}} P(o \mid S) * P(S \mid e)] * Compatibility(o,e)$$

In this formula, *S* represents the possible associated status of an object *o*. It could have three potential values: *G* (representing Gesture), *F* (Focus), and *D* (Display).

This function is determined by three components:

- The first, *P(o|S)*, is the *object selectivity* component that measures the probability of an object to be the referent given a status (*S*) of that object (i.e., gesture, focus, or visual display).
- The second, *P(S|e)*, is the *likelihood of status* component that measures the likelihood of the status of the potential referent given a particular type of referring expression.
- The third, *Compatibility(o, e)*, is the *compatibility* component that measures the semantic and temporal compatibility between an object and a referring expression.

Next we explain these three components in detail.

4.3.1 Object Selectivity

Given an object selected by a gesture (i.e., with a status Gesture), we currently use the approach described in [3] to calculate

P(o | S = Gesture). This measurement accounts for all the objects potentially selected by a gesture.

Given an object from the focus (i.e., not selected by any gesture), *P(o | S= Focus)* = 1/N, where N is the total number of objects that are in the focus vector. If an object is neither selected by a gesture, nor in the focus, but visible on the screen, then *P(o | S= Display)* = 1/M, where M is the total number of objects that are in the display vector. Note that each object to-be-considered is associated with only one of the three statuses. In other words, for a given object *o*, only one of *P(o|S=Gesture)*, *P(o|S=Focus)*, and *P(o|S=Visible)* is non-zero. Therefore, in terms of computation, the summation across different statuses for a given object is not actually applied.

4.3.2 Likelihood of Status

We use the data reported in [14] to derive the likelihood of the status of potential referents given a particular type of referring expression *P(S|e)*. We categorize referring expressions into the following six categories as in [14].

(1) Empty: no referring expression is used in the utterance.

(2) Pronouns: such as "it", "they", and "them"

(3) Locative adverbs: such as "here" and "there"

(4) Demonstratives: such as "this" "that", "these", and "those"

(5) Definite Noun Phrases: noun phrases with the definite article "the"

(6) Full noun phrases: other types such as proper nouns.

Table 1 shows the estimated *P(S|e)*. Note that, in the original data provided in [14], there is zero count for a certain combination of a referring type and a referent status. These zero counts result in zero probability in the table. We did not use any smoothing techniques to re-distribute the probability mass. Furthermore, there is no probability mass assigned to the status "Others". Basically, this probability table is completely based on the data reported in [14].

4.3.3 Compatibility Measurement

The *Compatibility(o, e)* measures the compatibility between an object *o* and a referring expression *e*. Similar to the compatibility measurement in [1], it is defined by a multiplication of many factors as follows:

$$Compatibility(o, e) = Id(o, e) * Sem(o, e) * \Pi_k Attr_k(o, e) * Temp(o, e)$$

In this equation:

***Id*(*o*,*e*)** captures the compatibility between the identifier (or name) for *o* and the identifier (or name) specified in *e*. It indicates that the identifier of the potential referent, as expressed in a referring expression, should match the identifier of the true referent. This is particularly useful for resolving proper nouns. For example, if the referring expression is house number eight, then the correct referent should have the identifier number eight. *Id*(*o, e)* = 0 if the identities of *o* and *e* are different. *Id*(*o, e)* = 1 if the identities of *o* and *e* are either the same or one/both of them unknown.

***Sem* (*o, e*)** captures the semantic type compatibility between *o* and *e*. It indicates that the semantic type of a potential referent as expressed in the referring expression should match the semantic type of the correct referent. *Sem* (*o, e)* = 0 if the semantic types of *o* and *e* are different. *Sem* (*o, e)* = 1 if they are the same or unknown.

Table 1. Likelihood of status of referents given a type of expression

P(S|E)	Empty	Pronoun	Locative	Demonstratives	Definite	Full
Visible	0	0	0	0	0.26	0.37
Focus	0.56	0.85	0.57	0.33	0.07	0.47
Gesture	0.44	0.15	0.43	0.67	0.67	0.16
Sum	1	1	1	1	1	1

***Attr*$_k$(*o, e*)** captures the domain specific constraint concerning a particular semantic feature (indicated by the subscript *k*). This constraint indicates that the expected features of a potential referent as expressed in a referring expression should be compatible with features associated with the true referent. For example, in the referring expression *the Victorian house*, the style feature is *Victorian*. Therefore, an object can only be a possible referent if the style of that object is *Victorian*. Thus, we define the following: $A_k(o, e) = 0$ if both *o* and *e* have the feature *k* and the values of the feature *k* are not equal. Otherwise, $A_k(o, e) = 1$.

***Temp*(*o, e*)** captures the temporal compatibility between *o* and *e*. Here we only consider the temporal ordering between speech and gesture. Specifically, the temporal compatibility is defined as follows:

$$Temp(o, e) = \exp(-|OrderIndex(o) - OrderIndex(e)|)$$

The order when the speech and the accompanied gestures occur is important in deciding which gesture should be aligned with which referring expressions. The order in which the accompanied gestures are introduced into the discourse should be consistent with the order in which the corresponding referring expressions are uttered. For example, suppose a user input consists of three gestures *g1, g2, g3* and two referring expressions, *s1, s2*. It will not be possible for *g3* to align with *s1* and *g2* to align with *s2*. Note that, if the status of an object is either Focus or Visible, then *Temp*(*o, e*) = 1.

This definition of temporal compatibility is different from the Gaussian function used in [1] that takes into consideration of real time stamps. From our empirical investigation on natural user behavior data, the new definition performs better than the previous Gaussian function when incorporated with the greedy algorithm.

4.4 An Example

Figure 3 shows an example of a complex input that involves multiple referring expressions and multiple gestures. Because the interface displays house icons on top of town icons, a point

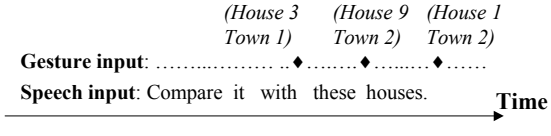

Figure 3: An example of complex input

Table 2: The Gesture Matrix (a) and Focus Matrix (b) for processing the example in Figure 3. Each cell in the *Referring Expression Matching* columns corresponds to an instantiation of the matching function.

Status	Potential Referent	Referring Expression Match	
		it	these houses
Gesture 1	House 3	1 × 0.15 × 1 = 0.15	**1 × 0.67 × 0.37 = 0.25***
	Town 2	1 × 0.15 × 0 = 0	1 × 0.67 × 0 = 0
Gesture 2	House 9	1 × 0.15 × 0.37 = 0.055	**1 × 0.67 × 1 = 0.67***
	Town 2	1 × 0.15 × 0 = 0	1 × 0.67 × 0 = 0
Gesture 3	House 1	1 × 0.15 × 0.14 = 0.02	**1 × 0.67 × 0.37 = 0.25***
	Town 2	1 × 0.15 × 0 = 0	1 × 0.67 × 0 = 0

(a) Gesture Matrix

Status	Potential Referent	Referring Expression Match	
		it	these houses
Focus	House 8	**1 × 0.85 × 1= 0.85***	

(b) Focus Matrix

(or circle) could result in both a house and a town objects. In this example, the first gesture results in both House 3 and Town 1. The second gesture results in House 9 and Town 2, and the third results in House 1 and Town 2. Suppose before this input takes place, House 8 is highlighted on the screen from the previous turn of conversation (i.e., House 8 is in the focus). Furthermore, there are eight other objects visible on the screen.

To resolve referents to the expressions *"it"* and *"these houses"*, the greedy algorithm takes the following steps:

1. The four input vectors, $\vec{g}$, $\vec{f}$, $\vec{d}$, $\vec{r}$ are created with lengths 6, 1, 8, 2, respectively.
2. A Gesture Matrix G_{62}, Focus Matrix F_{12}, and Display Matrix D_{82} are created.
3. These three matrixes are then initialized using the matching function described above. Table 2(a) shows the resulting Gesture Matrix. The probability values of $P(S|e)$ come from Table 1. The difference in the compatibility values for the house objects in the Gesture Matrix is mainly due to the temporal ordering compatibilities.
4. Next the GreedySortingGesture is executed. For each row in Gesture Matrix, the algorithm finds the largest legitimate value and mark the corresponding cell with *. Note that the corresponding cell for the row *i+1* has to be either on the same column or the column to the right of the corresponding cell in row *i*. These values are shown in bold in Table 2(a). Next starting from each column, the corresponding referring expression checks whether any "*" exists in its column. If so, those objects with "*" are assigned to the referring expressions based on the number constraints. In this case, since no specific number is given in the referring expression "*these houses*", then all three marked objects are assigned to "*these houses*".
5. After *"these houses"*, there is still *"it"* left to be resolved. Now the algorithm continuous to execute GreedySortingFocus. The Focus Matrix prior to executing the GreedySortingFocus is shown in Table 2(b). Note that since "*these houses*" is no longer considered, its corresponding column is deleted from the Focus Matrix. Similar to the previous step, the largest non-zero match value is marked (shown in bold in Table 2b) and assigned to the remaining referring expression *"it"*.
6. The resulting Display Matrix is not shown because it is not needed to resolve the referring expressions in this utterance.

Table 3: Performance comparison

	Input Type	Total Num	Graph Matching		Greedy	
			Num	%	Num	%
(a)	Total Inputs	219	129	58.9	134	61.2
(b)	Simple Inputs	186	117	62.9	119	64.0
	Complex Inputs	33	12	36.4	15	45.5
(c)	Correctly Recognized	127	103	81.1	106	83.5
	Incorrectly Recognized	92	26	28.3	28	30.4
(d)	Complex & Correctly Recog.	18	11	61.1	14	77.8
	Complex & Incor. Recog.	15	1	7.7	1	7.7

5. EVALUATION

To evaluate this approach, we use the 219 multimodal inputs collected previously [2] [2]. Table 3 shows the performance comparison between the greedy algorithm and the graph-matching algorithm. Overall, as shown in Table 3(a), the greedy algorithm performs comparably to slightly better than the graph-matching approach. Out of 219 inputs, the graph-matching algorithm achieves 58.9% accuracy and the greedy-algorithm achieves 61.2% accuracy. The major error sources for both algorithms come from poor speech recognition and language understanding, which were accounted for 55% and 20% of total errors respectively. Disfluencies are another problem. When a disfluency such as gesture repetition or repair occurs, the algorithm has no knowledge of such an exception and will mistakenly assign one or more objects from every gesture input to a referring expression. Thus disfluency detection will be helpful.

Table 3(b) and 3(c) show the detailed comparison along two different dimensions: the type of inputs and the performance of speech recognition. In both cases, the greedy algorithm performs comparably to the graph-matching approach. Table 3(d) further analyzes the difference in processing complex inputs. Out of 33 complex inputs, only 18 of them had expressions correctly recognized. Among those 18 correctly recognized complex inputs, there are three cases where the greedy algorithm works better than the graph-matching algorithm. The main reason is that parameters used in the graph-matching algorithm are very sensitive to the training process and sometimes parameters learned may not be generalized to find the best matches for the new inputs (as in the three cases). Since the greedy algorithm is guided by the

[2] The system used to collect the experimental data was developed with colleagues at IBM T. J. Watson Research Center when the first author worked at the Intelligent Multimedia Interaction group.

general cognitive and linguistic principles, it could potentially improve the robustness of interpretation.

6. CONCLUSION

We have described a greedy algorithm for efficient multimodal reference resolution that utilizes the linguistic and cognitive principles underlying human referring behavior. Our empirical studies indicate that this algorithm achieves comparable performance to the graph-matching algorithm for optimizing multimodal reference resolution. In particular, this algorithm relies on the theories of Conversation Implicature and Givenness Hierarchy to effectively guide the system in the matching process. Given *m* referring expressions and *n* potential referents from various sources, this algorithm takes *O(mn)* to find a solution. This is a dramatic improvement from the graph-matching approach that cannot be executed in polynomial time. Because of its simplicity and generality, this approach has a potential to improve the robustness of interpretation as indicated in our empirical studies. We have learned from this investigation that prior knowledge from linguistic and cognitive studies can be very beneficial in designing efficient and practical algorithms for enabling intelligent user interfaces. Our future work will combine the probabilistic reasoning with the prior linguistic and cognitive knowledge in one framework to address both adaptability and generality in input interpretation.

ACKNOWLEDGEMENT

This work was supported by grants from the National Science Foundation (IIS-0347548) and Michigan State University (IRGP-0342111). The authors would like to thank the anonymous reviewers for their helpful comments and suggestions.

REFERENCES

1. Chai, J., Hong, P., Zhou, M. X., and Prasov, Z. 2004c. Optimization in Multimodal Interpretation. In *Proceedings of ACL*, 2004, pp. 1-8. Barcelona, Spain.
2. Chai, J., Prasov, Z, and Hong, P. 2004b. Performance Evaluation and Error Analysis for Multimodal Reference Resolution in a Conversational System. *Proceedings of HLT-NAACL 2004 (Companion Volumn)*.
3. Chai, J., Hong, P., and Zhou, M. X. 2004a. A Probabilistic Approach to Reference Resolution in Multimodal User Interfaces, *Proceedings of 9th International Conference on Intelligent User Interfaces (IUI)*: 70-77.
4. Chai, J., Pan, S., Zhou, M., and Houck, K. Context-based Multimodal Interpretation in Conversational Systems. *Proceeding of 4th ICMI*, 2002.
5. Cohen, P., The Pragmatics of Referring and Modality of Communication, *Computational Linguistics*, 10, pp 97-146. 1984.
6. Cohen, P., Johnston, M., McGee, D., Oviatt, S., Pittman, J., Smith, I., Chen, L., and Clow, J. Quickset: Multimodal Interaction for Distributed Applications. *Proceedings of ACM Multimedia*, 1996. pp. 31– 40.
7. Grice, H. P. Logic and Conversation. In Cole, P., and Morgan, J., eds. Speech Acts. New York, New York: Academic Press. 41-58. 1975.
8. Grosz, B. J. and Sidner, C. Attention, intention, and the structure of discourse. *Computational Linguistics*, 12(3):175-204. 1986.
9. Gundel, J. K., Hedberg, N., and Zacharski, R. Cognitive Status and the Form of Referring Expressions in Discourse. *Language* 69(2):274-307. 1993.
10. Huls, C., Bos, E., and Classen, W. 1995. Automatic Referent Resolution of Deictic and Anaphoric Expressions. *Computational Linguistics,* 21(1):59-79.
11. Johnston, M, Cohen, P., McGee, D., Oviatt, S., Pittman, J. and Smith, I. Unification-based Multimodal Integration, *Proceedings of ACL'97*, 1997.
12. Johnston, M. Unification-based Multimodal parsing, *Proceedings of COLING-ACL'98*, 1998.
13. Johnston, M. and Bangalore, S. Finite-state multimodal parsing and understanding. *Proc. COLING'00*. 2000.
14. Kehler, A. Cognitive Status and Form of Reference in Multimodal Human-Computer Interaction, *Proceedings of AAAI'01*, 2000, *pp. 685-689*.
15. Oviatt, S. L. Multimodal interfaces for dynamic interactive maps. In *Proceedings of Conference on Human Factors in Computing Systems: CHI '96*, 1996, pp. 95-102.
16. Oviatt, S., DeAngeli, A., and Kuhn, K., Integration and Synchronization of Input Modes during Multimodal Human-Computer Interaction, In *Proceedings of Conference on Human Factors in Computing Systems: CHI '97*, 1997,
17. Oviatt, S. L. Mutual Disambiguation of Recognition Errors in a Multimodal Architecture. In *Proceedings of Conference on Human Factors in Computing Systems: CHI '99*.
18. Oviatt, S.L., Multimodal System Processing in Mobile Environments. In *Proceedings of the Thirteenth Annual ACM Symposium on User Interface Software Technology (UIST'2000)*, 21-30. New York: ACM Press.
19. Wahlster, W., User and Discourse Models for Multimodal Communication. *Intelligent User Interfaces, M. Maybury and W. Wahlster (eds.)*, 1998, pp 359-370.
20. Zancanaro, M., Stock, O., and Strapparava, C. 1997. Multimodal Interaction for Information Access: Exploiting Cohesion. *Computational Intelligence* 13(7):439-464.